

UNIVERSITÄT REGENSBURG



**Dual Use of Sensor Data –
Between Potential and Privacy**

DISSERTATION

zur Erlangung des Grades eines Doktors der Wirtschaftswissenschaft
eingereicht an der Fakultät für Wirtschaftswissenschaften der Universität Regensburg

Vorgelegt von:

Christian Roth, M.Sc.

`christian.roth@port17.de`

Berichterstatter:

Prof. Dr. Doğan Kesdoğan

Prof. Dr. Günther Pernul

Tag der Disputation:

15. Dezember 2022

Noah,
stay the curious explorer you are.

**

Acknowledgments

The endeavor was long and intense but shaped me as a person, both personally and professionally. I have to thank everyone who supported me during this experience of pursuing this ambitious research project, which got a bit out of hand at times.

First and foremost, I want to express my sincerest thankfulness to my doctorate supervisor, Doğan Kesdoğan, who has supported and encouraged me throughout this journey. He allowed me the freedom to explore my curiosity and scientific impulses but drew me back at the needed times with the appropriate questions and clues and ~~when required~~ outright challenged me. Without him, none of my scientific achievements would have been possible.

Secondly, I would like to thank my co-supervisor, Günther Pernul, who introduced me to the world of scientific work and sparked my interest in academia and research. Without him, I probably would not have chosen the university path after graduation.

Thirdly, I would like to express my gratitude to my colleagues at the Chair of IT-Security Management, with whom I was permitted to conduct scientific research. I have had innumerable intriguing and, most importantly, constructive debates over the years, especially with Mirja Nitschke. Additionally, my research group must be recognized for their tremendous contributions to this study effort. In particular, I want to thank Ngoc-Thanh Dinh, Marc Rossberger, and Markus Hornsteiner for what they did. Additionally, I would like to express my thankfulness to those students who took an interest in my subject area and delivered more than 100 notable results through their own seminars, bachelor's, or master's theses.

There are no adequate words to express my gratitude to my family. My endeavor has been a roller coaster. My wife Samra ensured that there was an up after every down. I am eternally indebted to her for her trust, motivation, and support. She had my back throughout the whole thing. And, of course, my son Noah, whose smile helped me keep going even on the hardest days. Many thanks to my parents, who have always ensured that I had an easy life without any major hurdles and have supported me with all their energy. In addition, my brother Michael, who set the bar high, and I had to follow (as it is between brothers), as well as his partner Teresa who was always ready to help. Last but not least, my sister-in-law Emina for taking the time to point out my sometimes peculiar choice of words.

This work is the result of and documents many years of intensive research. During this period, scientific issues have been addressed and published together with colleagues and students. Parts of this dissertation have already been published internationally at scientific conferences and in journals. In the sense of the scientific standard, this is indicated in appropriate places. It cannot be excluded that there are overlaps with these publications.

Christian Roth
Regensburg, June 2022

Contents

List of Figures	ix
List of Tables	xv
List of Listings	xvii
List of Sidebars	xix
List of Acronyms	xxi
List of Symbols	xxv
1 Introduction	1
1.1 Crowdsourcing, Crowdsensing and Mobility	3
1.2 Privacy and the Information Disclosure Triangle	5
1.2.1 Privacy Definition	5
1.2.2 Privacy Behavior	6
1.2.3 Data Economy	8
1.3 About this Work	13
1.3.1 Research Questions	13
1.3.2 Research Methodology	15
1.3.3 Structure of this Work	17
I Fundamentals	21
2 Sensor Data	23
2.1 Overview of Sensors and Sensor Data	24
2.1.1 Sensors in General	25
2.1.2 Types of Sensors	28
2.2 Location, Context Awareness and Motion Sensors	30
2.3 Quality of Sensor Data	33
2.3.1 Errors	34
2.3.2 Detection and Correction Methods	34

3	Privacy and Security Aspects of Sensor Data	39
3.1	Permission Workflow	39
3.1.1	Permissions	40
3.1.2	Declaring	41
3.1.3	Requesting	42
3.2	Discussion on Permissions	44
3.2.1	Organizational-based Privacy	44
3.2.2	User-based Privacy	45
3.2.3	Recapitulation	46
3.3	Example: The Bread (Joker) Malware Family	47
4	Alignr	51
4.1	Structured Literature Review	53
4.1.1	Research Questions	53
4.1.2	Search Process	53
4.1.3	Relevant Findings	54
4.2	Analysis of Speed Inference Methods	55
4.2.1	Overview of Methods (RQ1)	55
4.2.2	Deflections and Correction Methods (RQ2)	58
4.3	A Retrofittable Take on Speed Inference	65
4.3.1	Overview	66
4.3.2	Events	68
4.3.3	Coordinate Alignment	72
4.3.4	Rotation-based Gravity Removal	74
4.4	Proof-of-Concept Application	76
4.4.1	Architecture	77
4.4.2	API	79
4.4.3	Flow Overview	81
4.5	Implementation Flow	81
4.5.1	Data Layer	82
4.5.2	Middleware	84
4.5.3	Calculation Layer - Events	85
4.5.4	Calculation Layer - Finders	88
4.5.5	Integration Layer	91
4.5.6	Persistence Layer	95
4.5.7	Interface Layer	95
4.6	Evaluation	96
4.6.1	Gravity Removal	96
4.6.2	Coordinate Alignment	100
4.6.3	Accuracy of Speed Estimation	101

4.6.4	Performance Statistics	105
4.7	Conclusion, Error Sources, and Outlook	107
4.7.1	Overview of Literature	107
4.7.2	Our Approach	108
4.7.3	Error Sources	109
4.7.4	Outlook	110
5	Data Sets	111
5.1	Requirements	111
5.2	Collection	112
5.2.1	Process	112
5.2.2	Notation	114
5.2.3	Time-related Properties	116
5.2.4	Sensor Fusion	116
5.2.5	Additional Remarks	117
5.3	Presentation	117
5.3.1	Area	117
5.3.2	Example routes	120
5.4	Events	120
5.4.1	Extraction	124
5.4.2	Example	125
5.5	OpenStreetMap	128
5.5.1	Modeling	129
5.5.2	Quality Discussion	131
II	Privacy by Design-enabled Use Cases	135
6	Privacy by Design	137
6.1	Introduction	139
6.1.1	PETs	140
6.1.2	Privacy Aware System Design	141
6.1.3	Data Minimization	142
6.1.4	Designing Privacy by Design	144
6.1.5	Inclusive Design	146
6.2	Privacy-friendly Research	147
6.2.1	Overview of Research	147
6.2.2	Aspects of Research	153
7	STRIDE	157
7.1	Related Work	158

7.2	Vehicle-to-Anything	160
7.2.1	Requirements Towards V2X Architectures	162
7.2.2	Security and Privacy Building Blocks	163
7.2.3	V2X Infrastructure Standards	164
7.2.4	Malicious Behavior in V2X	164
7.3	<i>STRIDE</i> Architecture	168
7.3.1	Requirements	168
7.3.2	Privacy and the Integration of IFAL	169
7.3.3	Components	169
7.3.4	Separation of information between entities	172
7.3.5	Possible attacks on <i>STRIDE</i>	172
7.4	Simulation	174
7.5	Evaluation	177
7.5.1	Accuracy of the Network Utilization Reports	178
7.5.2	Estimation Robustness Against Dishonest Vehicles	181
7.5.3	Location Privacy-preservation Against Dishonest RSUs	182
7.6	Conclusion	187
8	ROADR	189
8.1	Related Work	190
8.2	ROADR Platform	192
8.2.1	Goals and Requirements	192
8.2.2	Architecture	193
8.2.3	Aggregation of Events	196
8.3	Mobile Companion	196
8.3.1	Collection and Preprocessing	196
8.3.2	Windowing	198
8.3.3	Detection	199
8.4	Overview of Events	200
8.4.1	Traffic Circles	200
8.4.2	Traffic Lights	203
8.4.3	Road Work	204
8.4.4	Road Quality	206
8.5	Detection of Events	209
8.5.1	Traffic Circle	210
8.5.2	Traffic Light	211
8.5.3	Road Work	213
8.5.4	Road Quality	213
8.5.5	Refactoring ML Models to TensorFlow Models	215
8.6	Evaluation	216

8.6.1	Preprocessing	216
8.6.2	Accuracy	219
8.6.3	Aggregation	223
8.6.4	Battery Consumption	225
8.7	Conclusion and Outlook	227

III Privacy Threats 229

9 Usage-Based Insurance: Pay-As-You-Drive and Pay-How-You-Drive 231

9.1	Overview of UBI rates in Germany	234
9.2	Meta-model of UBI schemes	235
9.3	Overview of premiums	238
9.3.1	Data Acquisition	239
9.3.2	Features	240
9.3.3	Processing	241
9.4	The Formulated Privacy Problem	241

10 Attacks on Sensor Data 247

10.1	Structured Literature Review	249
10.1.1	Research Questions	249
10.1.2	Search Process	249
10.1.3	Relevant Findings	250
10.2	Related Methodologies	253
10.3	Side-Channel Attacks in Literature	255
10.3.1	Overview of Attacks (RQ1)	255
10.3.2	Structure of Attacks (RQ2)	259
10.3.3	Clusters	263
10.4	Outlook on Protective Measures	264

11 Driver Identification 267

11.1	Structured Literature Review	269
11.1.1	Research Questions	270
11.1.2	Search Process	270
11.1.3	Relevant findings	271
11.2	Analysis of Driver Identification Methods	273
11.2.1	Data Acquisition (RQ1)	273
11.2.2	Workflow (RQ2)	276
11.2.3	Identification methods (RQ3)	278
11.3	Time Series-based Approach for Driver Identification	281
11.3.1	Features	283

11.3.2	Identification	288
11.3.3	Scoring	292
11.3.4	Decision	293
11.4	Evaluation	294
11.4.1	Preamble	294
11.4.2	Relevance of Features	295
11.4.3	Impact of Driver Set Size	296
11.4.4	Event significance	300
11.4.5	Learning rate	301
11.5	Realistic Environment	303
11.5.1	Moving to a Clustering Problem	304
11.5.2	Introducing New Driver	305
11.5.3	Non-uniqueness of Drivers	306
11.6	Conclusion	306
11.6.1	Overview of Literature	306
11.6.2	Our Approach	307
11.6.3	Privacy Implications	308
12	Reconstructing Trajectories Based on Sensor Data	311
12.1	Structured Literature Review	312
12.1.1	Research Question	312
12.1.2	Search Process	313
12.1.3	Relevant Findings	314
12.2	Analysis of Trajectory Inference Methods	315
12.2.1	Preconditions and Sensors (RQ1)	315
12.2.2	Summary of Approaches (RQ2)	318
12.3	Evolution of Trajectory Inference	322
12.3.1	Threat Model	323
12.3.2	Challenges	324
12.4	Building Blocks	325
12.4.1	Sensor Data	326
12.4.2	Model	326
12.4.3	Network Construction	328
12.5	Inferring Trajectories From Sensor Data	332
12.5.1	Retrieving Turn Candidates	333
12.5.2	Exploding Candidate Routes	334
12.5.3	Ranking of Candidate Routes	336
12.6	Evaluation	339
12.6.1	Setting	340
12.6.2	Sensor Accuracy	341

12.6.3	Success Rate	343
12.6.4	Impact of Filtering, Ranking, and Parameters	348
12.6.5	Feasibility	352
12.7	Conclusions	352
12.7.1	Overview of Literature	353
12.7.2	Our Approach	353
12.7.3	Privacy Implications	356
IV	Protection of Privacy	359
13	Overview of PETs in Sensor-Focused Environments	361
13.1	Structured Literature Review	362
13.1.1	Research Questions	363
13.1.2	Search Process	363
13.1.3	Relevant Findings	364
13.2	Results	365
13.2.1	Design Principles (RQ1)	366
13.2.2	Building Blocks (RQ2)	369
13.3	Conclusion	372
14	kUBI: Aligning Privacy and Integrity	375
14.1	Related Work	376
14.2	Framework	377
14.2.1	Stakeholders	378
14.2.2	Strategies	379
14.2.3	<i>kUBI</i>	380
14.2.4	Policies	385
14.3	Components	386
14.3.1	OS Domain	386
14.3.2	User Domain	388
14.3.3	Business Domain	394
14.3.4	Discussion on Design Principles	395
14.4	Evaluation	397
14.4.1	Environment	397
14.4.2	Anonymized Trip	398
14.4.3	Privacy Enhancement	398
14.4.4	Integrity Examination	402
14.5	Conclusion	404
14.5.1	The UBI Context	404

14.5.2	<i>kUBI</i> as a Balance	405
14.5.3	Privacy Implications	406
14.6	Outlook: Do We Need a TET Extension?	408
15	Recap with Discussion and Outlook	413
15.1	Research Questions	413
15.2	Summary of Results	415
15.3	Discussion and Outlook	419
	Appendix	425
A	List of Functions	425
B	Structured Literature Review	427
	References	429
	Supervised	469
	Images	471

List of Figures

1.1	Research published in relation with this dissertation	13
1.2	Design Science Research Methodology Process Model	16
1.3	Overview of the topics which are the subject of the dissertation including their relations	18
2.1	Example of the same event recorded continuously and discretely	24
2.2	Share of sensors build-in a smartphone released in and after 2020 after a market analysis	27
2.3	Overview of the coordinate system of vehicles and smartphones	29
2.4	Heatmap illustrating sensor combinations found in literature for the achieving user localization in indoor and outdoor scenarios	32
2.5	Methods applied to detect or quantify sensor errors organized by their frequency used	35
2.6	Methods applied to simultaneously detect and correct errors in sensor data streams	36
3.1	Overview of permissions requesting workflow	42
3.2	Overview of the three different kinds of permission request screens	43
4.1	Timeline with noteworthy publications in the field of velocity estimation based on Inertial Measurement Unit (IMU) sensor data	57
4.2	Taxonomy of building blocks that may be applied to the problem of velocity inference from sensor data	59
4.3	Applied building blocks within the proposed approach	66
4.4	Overview of the design of <i>alignr</i>	67
4.5	Example of a trajectory with a standing phase	68
4.6	Example of a trajectory with four turns	70
4.7	Relationship between the centripetal force, the tangential speed and the speed of a vehicle	70
4.8	Example of a trajectory with a road defects	72
4.9	Technical overview of the implementation of the <i>alignr</i>	78
4.10	Relationship between the lateral acceleration (i.e. centrifugal forces) when passing through a turn and the rotation around the z-axis	92

4.11	Example of the acceleration along the y-axis	97
4.12	Example of the impact of rotation on gravity	98
4.13	Example of drift acting on the gyroscope	99
4.14	Illustration of the delay to provide coordinate alignment	100
4.15	ECDF plot depicting the delta between the ground truth (i.e. Global Positioning System (GPS)) and the <i>alignr</i> -based speed values	101
4.16	Comparison of the speed estimation of <i>alignr</i> and the ground truth speed of the GPS	103
4.17	Illustration of the distances derived from <i>alignr</i> and GPS	106
4.18	Selected performance statistics for crucial operations of <i>alignr</i>	107
5.1	Distribution of trips recorded with a specific device $sp \in \mathcal{SP}$	118
5.2	Binned distribution of the trajectory length	119
5.3	Heatmap of the covered within the data set	120
5.4	Selection of exemplary routes	121
5.5	Relationship between Complex Event Processing and events	123
5.6	Exemplary route including multiple turns and curvy parts	126
5.7	Illustration of the heading changes of the traveled route	127
5.8	Distribution of the heading change error across multiple measurements	128
5.9	Exemplary map excerpt showing that a network is uniquely defined by nodes and edges	131
5.10	Example of incorrect modeling of the road width	132
5.11	Example of simplified modeling of intersections	133
5.12	Example of the deviation between model and real-world data	134
6.1	Basic architecture of an information system	138
6.2	Extended architecture of an information system with a Trusted Third Party	139
6.3	Relationship of different groups of harmful activities to the privacy of a user	140
6.4	Privacy design strategies are composed of methods that target the data itself and the process	145
6.5	Research published in the field of Privacy by Design	148
7.1	A Vehicle-to-Anything (V2X) network is composed of heterogeneous entities	160

7.2	Components of the European Telecommunications Standards Institute (ETSI) Intelligent Transport System (ITS) Security Certificate Management System architecture as per ETSI TS 102 941 and the <i>STRIDE</i> extensions	165
7.3	Classification of different attacks threatening V2X architectures	167
7.4	Vehicles move along predefined, realistic paths in the InTAS road network, which reflects a medium-sized city with industry and old town	175
7.5	Average activity of vehicles per hour throughout the simulation	178
7.6	Vehicles able to send during the simulation for different levels of k	179
7.7	Reachability of vehicles of the course of the simulation for different d_{send}	180
7.8	Comparison of speed derived by macroscopic view of all active vehicles and based on the Traffic Server (TS)'s aggregated view from traffic reports by vehicles	180
7.9	Divergence of speed for different ζ and amounts of attacking vehicles	182
7.10	Trajectory of a single vehicle illustrating the pseudonym change every $t_{i,val}$	183
7.11	Vehicles moving within the street network with specific vehicles highlighted	186
8.1	The <i>ROADR</i> platform is built on two pillars and four stakeholders	194
8.2	User-Interface of the <i>ROADR</i> application	197
8.3	Multi-level workflow within <i>ROADR</i>	198
8.4	Exemplary traffic circle with four exits	201
8.5	Pattern for traffic circle exit at 180° (2nd)	201
8.6	Patterns for traffic circle exits at 90° (1st) and 270° (3rd)	202
8.7	Patterns for a right turn and a traffic circle passing without countersteer resulting in a pattern similar to a right turn	203
8.8	A traffic light passing has three different phases	204
8.9	Overview of different types of road works	205
8.10	Overview of the phases of a road work passing	206
8.11	Relationship between parameters of the window functions and the respective traffic circle exit	210
8.12	Relationship between parameters of the window functions and the respective traffic light event lengths	212
8.13	Example of a Neural Network with one hidden layer	216
8.14	Excerpt of traffic events at the time of evaluation for the respective area of Regensburg, Germany	217

8.15	Pattern for a traffic circle that is divided into five sliding windows with a windows size ω of 20 s and an overlap o of 60 %	218
8.16	Road quality calculated by the <i>ROADR</i> application for all trajectories in the data set	222
8.17	Different interpretations of the same traffic circle	224
8.18	Trajectories of four different recording for the same traffic circle	224
8.19	Five different passing of a single road work	225
8.20	Overview of battery impact of <i>ROADR</i> on different devices . .	226
9.1	Distribution of reasons for accidents in Germany between 2010 and 2020	232
9.2	Overview of variants of Usage-Based Insurance (UBI)	233
9.3	Derived meta-model of UBI after analyzing twelve premiums .	235
9.4	Overview of PHYD process to derive a monthly premium	237
9.5	Telematic dongle offered by insurance company HUK-COBURG	239
9.6	Overview of risk indicators in UBI insurance schemes	240
9.7	Distribution of the applied methods of the UBI products and the origin of the data processor	242
9.8	Process of the privacy invasion of UBI products	242
10.1	Overview of side-channel attacks in the sensor-based setting . .	248
10.2	Treemap of side-channel attacks identified in the SLR	255
10.3	Distribution of sensor for a specific attack	259
10.4	Cluster map illustrating the used sensors by a specific attack vector	261
10.5	Sankey diagram illustrating the relation between sensors, attack classes, and attack vector	262
10.6	Clusters formed by the number of involved sensors	263
11.1	Quantitative metrics of the data sets that are used within the document corpus	274
11.2	Overview of sensor data used in different publications	275
11.3	Meta model of driver identification process	277
11.4	Overview of events used for driver identification	278
11.5	Example of the alignment of two sequences X and Y	283
11.6	Exemplary acceleration events for a driver in the data set	284
11.7	Illustration of an event across sensors	285
11.8	Comparison of maneuvers from different drivers and intensities	286
11.9	Example of the <i>DTW barycenter average</i> of multiple time series .	287
11.10	Exemplary trip with identified events	288
11.11	Overview of the kNN architecture	289

11.12	Overview of one acceleration event from each driver	291
11.13	Overview of one acceleration event from each driver	291
11.14	Overview of the achievable accuracy for assigning an event to a driver	295
11.15	Overview of the number of correctly identified trips as a function of test size and number of events	297
11.16	Accuracy for detecting a single event depends on the set size	298
11.17	Accuracy for correctly identifying a the driver of a trip	299
11.18	The confidence of a identification guess increases with longer trajectories providing more events eventually analyzed	300
11.19	Overview of the learning rate of the algorithm with different amounts of training data available	302
11.20	Silhouettes scores for a different number of clusters	305
12.1	Requirements and preconditions needed by the analyzed approaches	317
12.2	Overview of different approaches found in literature to select the final set of route candidates	320
12.3	Example path \mathcal{P} with $n = 4$ path events p_i	327
12.4	Section of the processed road network	329
12.5	Different challenges present in G	331
12.6	Illustration of difference between atomic heading changes and the ongoing curvature	338
12.7	Overview of the determination of different thresholds	341
12.8	Overview of three different area sizes used for the evaluation	343
12.9	Success rate for the trajectory inference attack across all test areas Q1-3	344
12.10	ECDF plot illustrating the percentage of exact matches within different amounts of candidates in dependence of the number of turns	345
12.11	Polar plots depicting the heading distribution of the different areas	346
12.12	Polor plots depicting the heading (or bearing) distribution of the different areas used in evaluation	347
12.13	Comparison of success rate for an exact match with related works	348
12.14	Success rate for the trajectory inference attack across all test areas Q1-3	349
12.15	Success rate for the trajectory inference attack across all test areas Q1-3	350
12.16	Impact of different settings for thresholds applied during database creation when identifieing turns	351

13.1	Proposals in the literature can be roughly divided according to the implementation approach	366
13.2	Overview of different requirements that the different approaches pose	367
13.3	Overview of the interactiveness of methods found in the literature	368
13.4	Distribution of the addressed entities that are to be protected by the privacy-enhancing technology	370
14.1	Layers of the Android Sensor Stack	382
14.2	Framework for gathering and processing sensor data from users in a corporate domain in a privacy-enhanced way for existing business models	383
14.3	$\vec{\mathcal{X}}$ are organized in \mathcal{B} and contain patterns representing \mathcal{E}_c . . .	389
14.4	Reference events $\check{\mathcal{E}}_{cil}$ with $i = 1, \dots, \nu = 3$ are representative for a category	390
14.5	Excerpt of a trip from a driver	399
14.6	Plot of the correlation between the number of events and the correctly recognized drivers	401
14.7	Example of the challenge to correctly select a reference event .	403

List of Tables

4.1	Overview of the 13 publications identified in the SLR	55
4.2	Relationship of approaches for speed inference and the specific sensors used as a data source	56
4.3	Identified errors in literature	60
4.4	Relationship between errors and the specific building blocks . .	64
4.5	Descriptive statistics of all trips used for the evaluation.	102
4.6	Descriptive statistics illustrating the impact of events for speed estimation	103
5.1	Overview of mobile devices <i>SP</i> used to create the data set . . .	113
6.1	Mapping of publications in the field of Privacy by Design and their applied privacy design strategies	154
7.1	Overview of default parameters that are used for the simulation	176
7.2	The ability of an attacker to successfully link pseudonyms based on timing and location is dependent on the enforced k or the sender range of vehicles	184
7.3	Probability of an attacker to successfully identify a random vehicle in two subsequent timestamps if a pseudonym change occurred	184
8.1	Overview of the 48 publications identified in the SLR	207
8.2	Performance of detecting a traffic circle including the correct exit	219
8.3	Performance of detecting traffic lights	220
9.1	Overview of the premiums analyzed in the survey offered by German insurance companies	238
10.1	Overview of the 61 publications identified in the SLR	251
10.2	Assignment of the individual publications to the respective attack class. The dominant class of attacks is keylogging with most works addressing this topic.	256
11.1	Overview of the 20 publications identified in the SLR	272

11.2	Matrix illustrating the different approaches found in literature including features used for identification	279
11.3	Overview of a trip evaluation to eventually predict a driver . . .	292
11.4	Overview of hyperparameter options and selected values for the identification approach	294
11.5	Confusion matrix for driver identification after twelve trips each	299
12.1	Overview of the 15 publications identified in the SLR	314
12.2	Relationship of approaches for trajectory reconstruction and the specific sensors used as a data source	315
12.3	Overview of default parameters that are used for the attack . . .	340
13.1	Overview of the 33 publications identified in the SLR	364
13.2	Matrix illustrating the combinations of the data retention and the applied privacy-preserving method across the document corpus	370
14.1	Overview of a trip evaluation to eventually predict a driver based on anonymized events as presented	400
14.2	Confusion matrix for driver identification after twelve trips each	402

List of Listings

3.1	Example of the <code>AndroidManifest.xml</code> (truncated)	41
4.1	Illustration of accessing sensor data	80
4.2	Exemplary <code>AlignrViewModel</code>	82
4.3	Exemplary <code>AlignrFragment</code>	83
5.1	Exemplary extract of recorded measurements \mathcal{M} illustrating the shape of the sensor data	118
5.2	Exemplary structure of a node	129
5.3	Exemplary structure of a way	130
5.4	Exemplary structure of a relation	130
7.1	Example output of Floating Car Data (FCD) data	177
14.1	Snippet of the application of the privacy-enhanced <code>SensorManager</code>	407

List of Sidebars

A	Floating Car Data and Floating Phone Data	26
B	Series Notation	114
C	A primer on Complex Event Processing (CEP)	122
D	Components of ABC4Trust	195
E	Driver Classification vs Driver Identification	268
F	A brief explanation of Dynamic Time Warping	282
G	Barycenter to average time series	287
H	Android Stack	381

List of Acronyms

A		E	
AA	Authorization Authority	EA	Enrolment Authority
ADAC	German Automobile Club	EaaS	Everyone-as-a-Sensor
ADAS	Advanced Driver Assistance Systems	ECU	Electronic Control Unit
AI	Artificial Intelligence	ETSI	European Telecommunications Standards Institute
ANN	Artificial Neural Network	F	
API	Application Programming Interface	FCD	Floating Car Data
		FN	False Negative
C		FP	False Positive
CAM	Cooperative Awareness Messages	FPD	Floating Phone Data
CAN	Controller Area Network	G	
CDP	Centralized Data Processor	GDPR	General Data Protection Regulation
CEP	Complex Event Processing	GNSS	Global Navigation Satellite System
CRL	Certificate Revocation List	GPS	Global Positioning System
D		GUID	Globally Unique Identifier
DENM	Decentralized Environmental Notification Message	H	
DIN	Deutsches Institut für Normung	HAL	Hardware Abstraction Layer
DTW	Dynamic Time Warping	HMI	Human-Machine Interaction
		HTML	Hypertext Transfer Markup

HTTP	Hypertext Transfer Protocol	OEM	Original Equipment Manufacturer
I		OHA	Open Handset Alliance
IFAL	Issue-First Active-Later	OS	Operating System
IMU	Inertial Measurement Unit	OSI	Open Systems Interconnection
IoT	Internet of Things	OSM	OpenStreetMap
ISO	International Organization for Standardization	P	
ITS	Intelligent Transport System	PATP	Pay-At-The-Pump
ITS-S	Intelligent Transport System Station	PAYD	Pay-As-You-Drive
K		PET	Privacy Enhancing Technology
kNN	k-Nearest-Neighbour	PHYD	Pay-How-You-Drive
L		PKI	Public-Key Infrastructure
LBS	Location-based Service	PoC	Proof-of-Concept
LSTM	Long short-term memory	PPM	Pay-Per-Mileage
M		R	
ML	Machine Learning	RFC	Random Forest Classifier
MVVM	Model View ViewModel	RQ	Research Question
N		RSU	Road-Side Unit
NN	Neural Network	S	
O		SDK	Software Development Kit
OBD	On-Board Diagnostic	SIM	Subscriber Identity Module
		SLR	Structured Literature Review
		SMS	Short Message Service
		StVO	Road Traffic Regulations (Straßenverkehrs-Ordnung)
		SVM	Support Vector Machine

T		UBI	Usage-Based Insurance
TEE	Trusted Execution Environment	V	
TET	Transparency Enhancing Technology	V2I	Vehicle-to-Infrastructure
TN	True Negative	V2V	Vehicle-to-Vehicle
TP	True Positive	V2X	Vehicle-to-Anything
TS	Traffic Server		
TTP	Trusted Third Party	W	
TÜV	Technischer Überwachungsverein	WAP	Wireless Application Protocol
U			

List of Symbols

Fundamentals

The following symbols may be reused and extended throughout this work

t	Timestamp	[ms]
\mathcal{T}	Set of multiple timestamps	[ms]
d	Distance	[m]
\mathcal{D}	Set of multiple distances	[m]
p	Propability as calculated using any probability method P	—
θ	Threshold either absolute or relative	—
λ	Weight either normalized or not	—
χ	Score, rating or confidence	—
m	Measurement of different shape	—
\mathcal{M}	Set of mutiple measurements	—
v	Velocity of an object	[m s ⁻¹]
acc	Accelerometer readings as a 3-dimensional vector	[m s ⁻²]
gyr	Gyroscope readings as a 3-dimensional vector	[rad s ⁻¹]
mag	Magnetometer readings as a 3-dimensional vector	[μ T]
gra	Gravity readings as a 3-dimensional vector	[m s ⁻²]
loc	Location composed of latitude and longitude	[$^{\circ}$]
lat	Latitude coordinate	[$^{\circ}$]
lon	Longitude coordinate	[$^{\circ}$]
h	Orientation of an object	[$^{\circ}$]
ω	Length of sliding window with varying unit	[m/s]
o	Overlap of two adjacent sliding windows	—

σ	Standard Deviation as calculated with <code>std</code>	—
----------	--	---

Sensor Data

The following symbols are introduced in Chapter 2

f	Frequency used to gather sensor data	[Hz = s ⁻¹]
-----	--------------------------------------	-------------------------

Alignr

The following symbols are introduced in Chapter 4

χ_g	Confidence of a speed guess	—
g	Speed guess having a timestamp and a speed estimation	[m s ⁻¹]
g^*	Speed guess that will be selected	[m s ⁻¹]
\mathcal{G}	Set of speed guesses	[m s ⁻¹]
\mathcal{G}^*	Set of speed guesses matching certain properties	[m s ⁻¹]
t_{ref}	Period since the last reference point was found	[s]
δv_{ref}	Constant that defines the window of a new speed guess	[s]
$\theta_{v,j}$	Threshold that defines a jump start	[m s ⁻¹]
$\theta_{v,t}$	Threshold that defines the maximum age of the speed guess to be selected	[s]
$\theta_{S,\sigma}$	Threshold that defines the transition between moving and standing	[m s ⁻²]
$\theta_{S,l}$	Threshold that defines the minimum length of a standing phase	[s]
$\tau_{B,l}$	Constant that defines the maximum considered autocorrelation lag	—
ω_B	Length of the sliding window for bump analysis	[s]
o_B	Overlap of the sliding window for bump analysis	[s]
$\theta_{B,\sigma}$	Threshold that defines points of interests for potential bumps	[m s ⁻²]
p	Peak representing a bump to be further analyzed	[m s ⁻²]
\mathcal{P}	Set of peaks	[m s ⁻²]

\mathcal{R}_p	Autocorrelation value calculated for two peaks	—
$\theta_{B,l}$	Threshold that differentiates real bumps from noise based on the autocorrelation value	—
w	Tangential velocity when passing a turn	[rad s ⁻¹]
r	Radius of a turn	[rad]
\mathcal{R}	Rotation matrix	—
$\theta_{C,gyr}$	Threshold that defines the minimum rotation of a turn based on the gyroscope	[rad s ⁻¹]
$\theta_{A,r}$	Threshold that defines the minimum force measured by the accelerometer in a turn	[m s ⁻²]
q	Quaternion	—
ϕ	Type of a sensor (e.g. accelerometer)	—
$\tau_{C,d}$	Constant that defines the number of ignored turn readings after initial detection	[s]
ω_C	Length of the sliding window for turn analysis	[s]
$\theta_{C,l}$	Threshold that defines the minimum force of the linear acceleration vector when evaluating forward acceleration	[m s ⁻²]
$\theta_{C,gyr}$	Threshold that defines the maximum rotation measured by gyroscope when evaluating forward acceleration	[rad s ⁻¹]

Data Sets

The following symbols are introduced in Chapter 5

p	Person, object or entity of some kind	—
\mathcal{P}	Set of persons, objects or entities	—
sp	Smartphone, for instance, used to gather sensor data	—
\mathcal{SP}	Set of smartphones	—
G	Structure representing a graph	—
v	Vertex found in a graph	—
V	Set of vertices found in a graph	—
e	Edge found in a graph	—

E	Set of edges found in a graph	—
a	Ongoing driving activity such as moving or standing	—
\mathcal{A}	Set of driving activities	—
e_a	Atomic event representing an outstanding pattern in a data stream	—
\mathcal{E}_a	Set of atomic events	—
e_c	Complex event representing an outstanding and long-lasting pattern in a data stream	—
\mathcal{E}_c	Set of complex events	—

STRIDE

The following symbols are introduced in Chapter 7

l	Driving lane belonging to a street found in a road network	—
$\Delta\mathcal{T}$	Length of a discrete timeslot	[min]
\mathcal{I}	Permanent identify of a vehicle	—
p	Temporary pseudonym of a vehicle	—
\mathcal{P}	Set of temporary pseudonyms of a vehicle	—
z	Certificate assigned to a temporary pseudonym	—
\mathcal{Z}	Set of certificates assigned to a temporary pseudonym	—
d_{send}	Range within other participants can be reached for sending and receiving messages	[m]
θ_{send}	Probability that a participant will send in a time step	—
k	Minimum size of an anonymity set that has to be ensured	—
$t_{t,val}$	Duration of the IFAL pseudonyms' validity	[min]
$t_{t,o}$	Overlapping time frame where two IFAL pseudonyms of the same identity are valid	[min]
κ	Proportion of dishonest participants that will lie	—

ζ	Applied method to pollute speed information when lying	—
ζ_t	Time frame where messages of the same pseudonym are discarded	[min]

Driver Identification

The following symbols are introduced in Chapter 11

k	Number of neighbors to consider in the k-Nearest-Neighbour algorithm	—
ω	Weight distribution of neighbors	—
η	Metric to use when computing Dynamic Time Warping (DTW) similarity	—

Reconstructing Trajectories Based on Sensor Data

The following symbols are introduced in Chapter 12

\mathcal{P}	Path that formally describes the relative movement of a vehicle	—
p	Path event as a single driving event composing a path	—
\mathcal{G}	Street network being an extended street graph G	—
s	Road segment that represents edges e in a street network	—
\mathcal{S}	Set of road segments	—
c	Street connection between two road segments with particular properties	—
\mathcal{C}	Set of street connections	—
\hat{c}	Road connection that represents a turn	—
$\hat{\mathcal{C}}$	Set of all turns found in a street network	—
$\tilde{\mathcal{C}}$	Set of all straight connections found in a street network	—
r	Route that represents a specific combination of edges and	—

r^*	Route candidate that meets certain requirements	—
\mathcal{R}^*	Set of route candidates	—
θ_S	Threshold that defines a slight turn	[°]
θ_A	Threshold that defines the maximum angle error	[°]
θ_S	Threshold that defines the maximum, relative distance error	—
τ_s	Threshold that defines the absolute distance error	[m]
θ_M	Threshold that defines the maximum compass error	[°]
θ_H	Threshold that defines the maximum heading error	[°]
λ_A	Weight of the angle similarity score	—
λ_D	Weight of the distance similarity score	—
λ_H	Weight of the heading similarity score	—
λ_C	Weight of the curvature similarity score	—
λ_{TL}	Weight of the traffic light positional score	—
χ_Σ	Score of a route candidate r^*	—
χ_A	Score defining the turn angle similarity between \mathcal{P} and r^*	—
χ_H	Score defining the heading similarity between \mathcal{P} and r^*	—
χ_D	Score defining the distance similarity between \mathcal{P} and r^*	—
χ_C	Score defining the curvature similarity between \mathcal{P} and r^*	—
χ_{TL}	Score defining the turn angle similarity between \mathcal{P} and r^*	—

kUBI: Aligning Privacy and Integrity

The following symbols are introduced in Chapter 14

γ_{e_c}	Category of a complex event as assessed by C	—
Γ_{e_c}	Sequence of event categories as assessed by C	—

$\tilde{\Gamma}_{e_c}$	Set of event categories provided by a third party (reference set)	—
μ	Number of disparate event categories	—
ν	Number of events per event category	—
ρ	Resampling rate, i.e. number of elements in a sequence after processing through R	—
Ψ	Policy defining any kind of rules such as properties or requirements	—
\vec{x}	Generic sensor reading from an unspecific sensor	—
$\vec{\mathcal{X}}$	Set of generic sensor readings	—
b	Data block containing multiple generic sensor readings	—
\mathcal{B}	Set of data blocks	—
\hat{b}	Processed data block to provide anonymity	—
\mathcal{B}'	Set of data blocks that are logically related (within a trip)	—
$\widehat{\mathcal{B}'}$	Set of data blocks that are logically related and anonymized	—
\hat{e}_c	Processed complex event block to provide anonymity	—
$\hat{\mathcal{E}}_c$	Set of anonymized complex events	—
\mathcal{E}'_c	Set of complex events that are logically related (within a trip)	—
$\check{\mathcal{E}}_c$	Set of complex events provided or used by a third party	—
σ	Cryptographic signature	—
Σ	Set of cryptographic signatures	—
d	Private key used for asymmetric encryption or signature schemes	—
c	Public key used for asymmetric encryption or signature schemes	—

Introduction

1

[Privacy is] the right to be let alone — Warren and Brandeis [395]

This statement of Warren and Brandeis from 1890 sounds antiquated in the days of social networks, smart cities, or connected cars. In fact, rapidly evolving technologies, including but not limited to the Internet, have led to a permanent and ubiquitous exchange of information, data, and knowledge not only by individuals but also by intelligent devices. Nevertheless, privacy is not an outdated and forgotten concept; in fact, it is more relevant than ever.

George Orwell's dystopian novel *1984* [287] paints a picture of a totalitarian surveillance state in which the populace is subdued by propaganda and perfect control. It is the goal of the government to completely repress the population's manner of life as well as their way of thinking, which is accomplished through suitable propaganda and severe instruments of punishment. State-prescribed doublethink, as characterized in the book, is the perception of reality that is subjected to circumstantial and personally identifiable information-invading inspection by the state. When it comes to the fundamental right to privacy, Orwell's fiction is the polar opposite of Warren and Brandeis's assertion of that right in a society where free expression is crucial. An essential tool for oppression is the equalization of privacy regardless of the situation because persons thereby live in fear and must comply fully with the principles set forth. As a result, this is the most crucial protectable good in our democratic society because it ensures that each individual may exercise his own autonomy.

The publications surrounding global surveillance disclosures beginning in 2013, in which countries from all over the world participated in the comprehensive and suspicionless collection of citizens' private information, demonstrate the unintentional invasion of privacy that has occurred in recent years. Because of the incident, it has been demonstrated that fiction may become a reality and that individuals can be widely watched without their knowledge or agreement.

As a counterbalance to this covert violation of privacy, though, overt systems exist in which individuals appear to be participating willingly and in their own best interests. In addition to its state monitoring measures (i.e. the Great Firewall), China is using the so-called Social Credit System [50, 225] to inspire and enforce desirable conduct among its populace [78], an effort that can be compared to

Violating
privacy

Extrinsically
motivated
monitoring

1984's doublethink and censorship. Officially, the system intends to promote the trustworthiness and morals of residents and businesses by automatically analyzing and grading their financial and social behavior [86]. However, there are significant parallels to Orwell's novel here, as users become transparent, and those who resist facing apparent penalties. Due to the ramifications and restrictions on public and private life, there is no longer any discussion about voluntary participation in this program.

Ubiquitous
computing
against
privacy

Again, it is possible to claim that the system is in use due to governmental compulsion. We may also find parallels to George Orwell once more. Unfortunately, there is no easy way out of the privacy quagmire. However, loss of privacy does not always have to be extrinsic (motivated). Weiser [397] describes a vision in which the sharing of information constantly occurs, without barriers, delay, or the need for considerable input from the user, using the term *ubiquitous computing*. This vision appears to be present nowadays, and technological advancements and increased connectivity enable it. Users have the ability to share their lives in their entirety continuously, and the lines between the actual and virtual worlds, between those who are physically there and those who are electronically linked, and between forgetting and remembering are blurred. While the information in the preceding instances had to be gathered by third-party services, the insights into one's personal life, including one's worldview, are offered by the users themselves and may be exploited, according to Chan [69]. Individual privacy appears to be a lost idea in this context; humans, as social beings, do not want "*to be let alone*", but rather to be ubiquitously in contact with others.

[W]henever we face privacy sensitive decisions, we hardly have all data necessary for an informed choice. But even if we had, we would be likely unable to process it. And even if we could process it, we may still end [sic] behaving against our own better judgment. — Acquisti [2]

Overburdened users

From the perspective of Acquisti, the attitude toward privacy appears to be different and is consistent with studies that have found that privacy is still essential for individuals. However, it appears that the complexity of information systems makes decision-making substantially more difficult. Because advancement is a motivating force for development, it is necessary to find a way to handle data in a balanced manner with the many players in these complex systems without overburdening single participants. The dissertation examines the dichotomy between the use of comprehensive personal data and the preservation of privacy to protect people and our democratic world as a whole.

Crowdsourcing, Crowdsensing and Mobility

1.1

Nowadays, crowdsourcing is a well-known concept in which a system owner-defined task is outsourced to a large group of people called a crowd (typically an online community [157]) in the form of an open call [51]. They are motivated by intrinsic or extrinsic factors [55], the online community can work collaboratively to complete the task, or individuals can carry it out alone. The term was coined by Jeff Howe and Mark Robinson in the June 2006 issue of *Wired* magazine [180] and was later formalized [179]. There are various advantages associated with crowdsourcing. Optimized costs for the contractor and increased execution speed can be assumed due to this disruptive concept. As a matter of fact, the size of a crowd is dynamic, which can imply greater flexibility, scalability, and quality at the same time [51, 55]. This is based on the fundamental idea that a collective can provide better and more comprehensive knowledge (“*wisdom of the crowd*”) compared to individuals or even experts and thus develop more sophisticated solutions [361].

Crowdsour-
ing

Crowdsourcing as a concept was mainly seen as a web-based process but is changing towards mobile devices [124]. It is essential that the term crowdsourcing is clearly distinguished from crowdsensing, which can be defined as “*a new sensing paradigm that empowers ordinary citizens to contribute data sensed or generated from their mobile devices, aggregates and fuses the data in the cloud for crowd intelligence extraction and people-centric service delivery*” [157]. In contrast, crowdsourcing is about harnessing humans’ knowledge, understanding, and intuitiveness when they are entrusted with the task. This is different from crowdsensing, which is about gathering detailed and holistic data to share it with a service for further processing. However, further processing may involve intelligent systems but can also be combined with human-driven processes such as crowdsourcing [157].

Crowdsens-
ing

With the rise and ubiquity of smartphones equipped with multiple sensors and connectivity, crowdsensing becomes convenient as no additional devices are needed to contribute data to online platforms [200]. Eventually, this is preferable from a monetary perspective since dedicated sensors are an additional cost component, require installation and maintenance, and are only applicable for one previously defined use case [136, 241]. Especially in the mobile setting, crowdsensing is favorable because it works in a decentralized way, with mobile sensors being the participants of the mobile network itself. Hence, every driver, cyclist, pedestrian, or passenger can contribute valuable, independent, and individual

Convenience
of crowd-
sensing

information, contributing to the growth of the crowdsensing network. They form a mobile, physical community [157]; hence the term mobile crowdsensing.

Participatory
sensing and
opportunistic
sensing

Participatory and opportunistic sensing are two forms found in the context of mobile crowdsensing, although most mobile crowdsensing approaches use a hybrid kind, including both types [157]. Participatory sensing is also called community sensing and relies on the active participation of everyday users using their own mobile devices to gather, analyze, and share the locally generated knowledge with a remote platform. For example, participatory users can distribute information on road accidents that they have recorded manually. To relieve the user from actively contributing data, opportunistic sensing uses the built-in sensors of mobile devices, such as smartphones, to collect data autonomously in the background. As an example, road condition measuring is a task that can be automated. Opportunistic sensing may also increase the objectivity of the submitted data as no user interpretation is required [420] and attract more participants due to the lower effort required.

A note on
privacy

Crowdsourcing and crowdsensing are related but different, although approaches that combine both principles are present. Let us present an example to discuss the relevant opportunities and challenges of crowd-based platforms. Waze¹ is a GPS-based route planning service now owned by Google that takes into account the current traffic situation and arbitrary events. The said events are provided by the community in a crowdsensed and crowdsourced fashion. Users can, for example, provide information on closed roads, while other participants can confirm location-related events. Attached to the submitted events (or other activities) are the long-term pseudonyms of a user who may also be visible when the road network is traversed. The barrier to entry through a smartphone app is low and such types of solutions scale well accordingly. Furthermore, when the critical mass of the community size is reached, economies of scale can be relied upon to draw a comprehensive and up-to-date picture of traffic events. From a privacy perspective, Waze enables not only the service provider itself to derive the movement patterns of single users based on location data but also third parties by cross-referencing events to users via publicly available information [136]. We conclude that it is of particular interest to protect the privacy of users while ensuring the integrity of the data to provide impeccable service quality eventually. Exactly this dichotomy is the subject of this dissertation.

¹<https://www.waze.com>

Privacy and the Information Disclosure Triangle

1.2

Modern applications of various social, governmental, or economic domains rely on the basis that data is shared and exchanged between different entities accordingly, with crowdsensing-based platforms being only one example. Depending on the application, the data contains sensitive information. Therefore, the disclosure of this information poses a threat to privacy. Negative consequences for the individual are conceivable [142], so actions must be well-considered and weighed.

In this section, we will first define the term privacy before discussing the constraint of the data sharing behavior. Although privacy and personal data are highly valued, people behave contrary to their attitude and share more data [284]. The causes are manifold and cannot be conclusively substantiated, yet this section will offer rationales that might explain behavior. For this purpose, we conducted a Structured Literature Review (SLR) to answer the question of what the antecedents of information disclosure are, the results of which are presented in this section [407].

Structure

Privacy Definition

1.2.1

According to the General Data Protection Regulation (GDPR), users have a right to privacy, i.e. they have the authority to decide who has access to process their data and for what reason. The term *privacy* is used in many ways and subject to a broad range of interpretations, some being vague, intangible, contradicting, or complex [350]. However, a clear understanding of its meaning is critical to making the appropriate decisions [142].

In the context of the GDPR and, in particular, this work, the definition of Westin [398] seems to be well-fitted. He defines privacy as “*the claim of individuals, groups or institutions to determine for themselves when, how, and to what extent information about them is communicated to others*”. Not only does he focus on personal information, but he also describes elements of information processing. Westin implicitly denotes that an individual can on his own, without justification, decide what information is made available to the public and what is kept private.

Definition of Westin

Laufer and Wolfe [226] define privacy by identifying three distinct dimensions, namely the self-ego dimension, the environmental dimension, and the interpersonal dimension. The taxonomy-based approach of Solove [350] aims to make the meaning of privacy more tangible by categorizing various forms of privacy violation accordingly. The categories are Information Collection, Information

Further relevant definitions

Processing, Information Dissemination, and Invasion. Smith et al. [348] classifies all current definitions into value-based definitions (general privacy as a right, privacy as a commodity) and cognate-based definitions (general privacy as a state, general privacy as control) in a thorough review of all relevant definitions to date.

Importance
of privacy

After having defined privacy, one needs to understand why privacy is important in the first place. Solove [351] tried to answer this particular question of privacy's value by defining tangible and comprehensible arguments why to care. In this opinion, privacy is fundamental for *limit on power, reputation management, maintaining appropriate social boundaries, trust, control over one's life, freedom of thought and speech, freedom of social-political activities, ability to change and have second chances, protection of intimacy, bodies, and sexuality and not having to explain or justify oneself.*

Privacy
concerns

Any disclosure of information can impact each listed protected right and aspect. The concern of (possibly) losing one's privacy as a consequence of information disclosure towards a specific external agent is defined as *Privacy Concern* [408]. It differs from previous perceptions that considered privacy concerns being a general concern like an attitude [348]. However, according to Xu et al. [408], it is a situation-specific privacy construct.

Willingness
to disclose

Another relevant but differently interpreted construct related to information disclosure and privacy is the so-called *willingness to disclose*. The wording is diverse; willingness to share or willingness to provide is also found in literature [142, 212]. Eventually, *willingness to disclose* can be considered as an "*individual's openness to the idea of providing specific personal information in the context of ecommerce transactions*" [311] i.e. willingness being a subject's attitude, intention, preference, or concern towards privacy-specific topics [284]. In contrast, willingness may sometimes be used actually to quantify a person's actual behavior [110].

1.2.2 Privacy Behavior

Privacy
paradox and
its evolution

Under the assumption that willingness to provide indicates the amount of information individuals are *willing* to share (behavioral intention), Norberg et al. [284] indicates a conflict that the attitude is not directly transferable to the actual behavior of a person. This effect is called *Privacy Paradox*. In fact, it was shown that there is a significant imbalance between the information provided and intended. The privacy paradox is hard to grasp due to numerous constructs (e.g. attitudes, concern, behavioral intention, and actual behavior) [284] and het-

erogeneity [142] in the field of privacy. The privacy paradox can be considered a construct itself with an input and an output variable. A broad range of input variables is conceivable, with Norberg et al. [284] proposing behavior intention or willingness to provide, while others [4, 110, 212] refer to i.a. intention, preference, attitude, and concern. Finally, the input variable causes the output variable but cannot fully explain or predict it. This is caused by the privacy paradox. The actual behavior would be expected to be the result (as in the case of Norberg et al.). However, the literature also proposes behavioral intention or willingness to provide [110, 212]. The latter is common in the privacy calculus theory. In Wurmer et al. [407], we, therefore, introduce the novel “Two-Step Privacy Paradox” where every transition yields a paradoxical gap that needs explanation.

One prominent approach to explain the privacy paradox is a theory called *Privacy Calculus*. The privacy calculus theory assumes that individuals do not make decisions objectively but weigh a trade-off between benefits (potential gain for disclosure) and costs/risks, i.e. the expected loss of privacy. It originates from the social exchange theory, where social interaction is only performed if the benefits outweigh the costs and risks [419], and was then adapted as a calculus of behavior, where people evaluate the possible future implications of their actions. However, this is difficult for an individual to achieve, as e.g. rules, settings, and technologies can change unpredictably in the future [226]. Culnan and Armstrong [88] concluded that procedural fairness reduces privacy concerns in the context of consumer transactions. Consumers see procedural fairness in how companies handle disclosed data. This implicates a consumer cost-benefit analysis [89].

Privacy
calculus

The Theory of Planned Behavior (and its predecessor, the Theory of Reasoned Action) states predictors of intention, namely attitude (beliefs about a behavior), subjective norm (beliefs about others’ attitudes toward a behavior), and perceived behavioral control (beliefs about one’s ability to perform a behavior) [5]. Attitude, subjective norm, and perceived behavioral control are, in this case, functions of the related underlying *belief* of an individual. However, multiple beliefs that influence intention can be contrary “that together the beliefs comprise a set of elements in a calculus or a decision process” [110]. Recent research suggests that benefits may outweigh costs in influencing behavioral intention [415]. Since beliefs can change, it may also be possible to influence a person’s actual behavior accordingly.

Beliefs of
individuals

1.2.3 Data Economy

This work focuses on data sharing, which ultimately includes sensitive information about individuals. As mentioned, the decision-making process of information disclosure is complex and influenced by different dimensions, including personal beliefs. It is viable and necessary to understand why people share data and how to answer the privacy paradox, or at least try to find explanations in the form of antecedents that may be coherent and multilayered [3]. For instance, when users disclose their information, it appears as if they do not care about privacy, but the process is more complex: users disclose large amounts of data because they are required to do so. Otherwise, they are denied participation in the process, which, in turn, is also impossible and unpleasant for users due to social constraints [216].

Various
factors
influencing
data sharing

Multiple attempts are made to categorize answers to the privacy paradox, including Acquisti et al. [3], Barth and Jong [35], Dinev et al. [111], Gerber et al. [142], Kokolakis [212], and Solove [351]. In the following, we try to present a common and integrative understanding considering the different directions and streams presented by several researchers. We argue that information disclosure may be explained by three different dimensions (of antecedents), which often stood out in the context of the literature study. First, the already introduced Privacy Calculus (to balance benefits and costs); second, the cognitive perception approach; and third context-based concepts. Together, they all compose a triangle where each aspect impacts the disclosure process. Eventually, the dimensions may influence each other, stressing a particularly complex decision process.

Benefits and Costs

The first dimension is benefits and costs going back to Dinev and Hart [110], who applied the concept of information disclosure to e-commerce, describing a phenomenon where specific benefits override concerns about the disclosure process. Consequently, the decision process is a trade-off between benefits (*what do I get?*) and costs (*what do I lose?*)². Acquisti [2] relates costs to cognitive perception indicating a reciprocal influence, further picked up by Gerber et al. [142]. Based on recent findings [348], we selected four properties (i.e. antecedents) for this dimension that are—according to our understanding—considered when balancing benefits and costs.

²Costs are not necessarily connected to a piece of specific information made available to a third party but can be long-term, multidimensional consequences such as identity theft or social stigmas.

Monetary reward There is broad consensus in the literature that a *monetary reward* influences the decision-making process where a direct compensation is given to the disclosing object for his data. Furthermore, implicit compensation, as found in loyalty programs, is a meaningful choice as a reward [301]. However, it is not possible to derive a general threshold for the amount of monetary reward that enables the disclosure of information in any case, as it depends on multiple variables [3]. Although most research addresses the correlation to the monetary reward, some argue that money explicitly implies a value in data resulting in a negative impact on the information disclosure process [229, 400].

Personalization Based on the personal information disclosed and learned, the process of *personalization* is to address an individual using specifically adapted communications or experiences such as an online shopping experience with interest-based recommendations. However, this antecedent is controversial, and there is no joint agreement. Some argue that it has positive consequences on the disclosure process (e.g. [188]), no impact at all (e.g. [202]), or even negative influences (e.g. [49]). The last relationship is interesting as it is similar to the negative consequences of offering a monetary reward with the user being made aware of how data may be exploited.

Convenience *Convenience* describes an increased level of comfort (e.g. less time or effort required) experienced by the user when interacting with an information system and, as such, is primarily preferable [110, 188].

Context-specific Depending on the current *context*, specific benefits may arise. However, the term context is different from the dimension called context, which describes the setting in which the decision of the information disclosures takes place. Sun et al. [359] differentiates between utilitarian and hedonic benefits, such as social benefits. We expect that gamification is also related to this context-specific antecedent. For example, gaining points for submitting information may motivate participants to be active within crowdsourcing systems.

Cognitive Perception

In principle, the decision-making process is determined by a person's fundamental privacy behavior, including the privacy calculus. In reality, as mentioned above, there is a distinction between action and attitude, which eventually results in the privacy paradox [2]. Biases and heuristics affect people's decision-making processes, whether consciously or unintentionally [110], which can result in a change in perspective (cognitive perception). Selected antecedents will be introduced below.

Incomplete/Asymmetric Information and Bounded Rationality According to the privacy calculus, each decision on information disclosure is rational and is made based on an (individual) trade-off of benefits and costs, resulting in optimal, self-determined behavior. However, an individual cannot make such a rational decision because it requires complete knowledge of a transparent data-consuming process. This is usually not the case, and there is an imbalance between user and service provider regarding the process. Either the user has only *incomplete knowledge* about the process, or the service provider has additional information on how the data will be used [2, 6] This problem is intensified by the unpredictability of future developments (either technical or event-wise) so that the costs of a decision that could still arise (e.g. information theft) would not be assessable. Even if a user had all information about the process, he would not be able to make the best possible and thus the most rational decision based on the information due to limited possibilities. This is described as *bounded rationality* [344]. One of the reasons for this is that the concept of benefits and costs is complex and challenging to grasp. Quantification for an unambiguous comparison is unrealistic, and consequently, so is the perfectly rational decision [2].

Hyperbolic Discounting and Immediate Gratification The time-inconsistent valuation of benefits can be explained using *hyperbolic discounting* effects and *immediate gratification* [2]. Hyperbolic discounting describes an effect in which users prefer or consider short-term (albeit smaller) benefits or costs at the expense of benefits or costs that lie in a more distant future. Applied to privacy, the information disclosure process can be influenced by granting a monetary reward in the short term, for example, even if costs incurred in the future (such as identity theft) would weigh more heavily. The drag of immediate gratification leads to a decision favoring the short-term reward. The threat to privacy is exacerbated by the difficulty of the tangibility of cumulative consequences that compound over time [2].

Framing Effects and Endowment Effect According to literature, the way an information disclosure process is presented (for example, the wording of a prompt) has a significant influence on the behavior of users. This is described as *framing effect* [29, 196]. Related to the framing effect is *endowment effect* [196]. The request for further information by a service provider to maintain membership in a social network is an example of the endowment effect, as the fear of losing an already acquired object (i.e. the membership) outweighs the desire to obtain it first (i.e. to become a member). Therefore, this effect affects the information disclosure process.

Optimistic Bias *Optimistic bias* describes an effect in which people frequently underestimate their chance of being the victim of a privacy infringement compared to dangers that other individuals face [80, 142]. Hence, no rational decision can be made when shifting objective information.

Nothing-To-Hide Another bias is skipping (or shortcutting) of an information-disclosure process by arguing that one has nothing to hide (especially from the government). According to Solove [350], the *Nothing-To-Hide* argument is a fallacy. Privacy is not comparable to secrecy which attempts to conceal a piece of information, but is about protecting personal interests and not about hiding a crime. Additionally, people are unaware that their data may be aggregated, exploited, insufficiently protected, or passed on for secondary purposes, eventually becoming something they would have intended to conceal in the first place [350].

Nudging, Dark Patterns, and Friction Deliberate manipulation or influence of the decision-making process by the requester is a way to induce the user to disclose data. Awareness of privacy and corresponding assertion of interests on the part of users are manipulated by design of technology, and especially user interfaces. Risks are minimized by design, and data exchange becomes natural. In this way, businesses can increase information sharing through *nudging* [351]. Because this design process reflects only the interests of the service provider, it is considered a *dark pattern* [388]. The information disclosure process is often deliberately made as convenient and direct as possible to minimize friction. On the contrary, *friction* for privacy-related topics is intentionally increased [351].

Context

Context is a third dimension that influences the decision process about information disclosure, as context is a significant aspect of how an individual perceives and values privacy [153, 281]. The difference in contexts between different disclosure processes may explain the gap between the basic attitude and actual behavior in specific situations [253]. In privacy literature, the element of context is often underrepresented and overlooked [3, 351]. The context is composed of a person and a situation. Although each antecedent is explained in detail in Wurmer et al. [407], this section only mentions selected elements important for the further progression of this work.

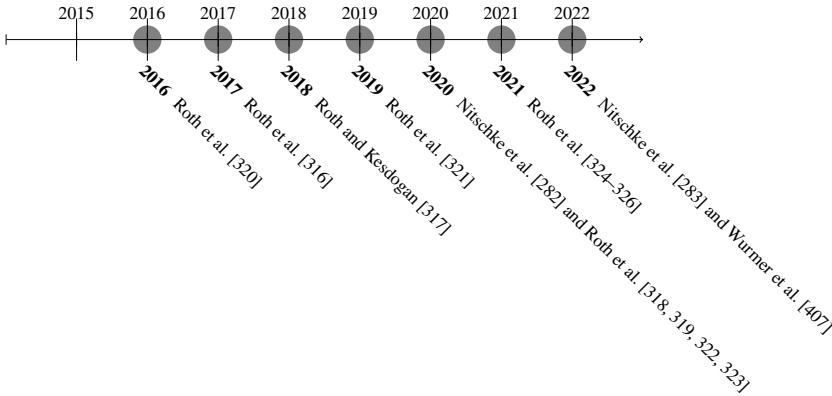
Personal Antecedents Basically, *personal* and objective characteristics such as age [4, 49, 191], gender [4, 49, 191, 430] and cultural background [214, 249, 311, 372] and education [311, 396] influence the information disclosure process. In summary, young people are generally more willing to share information [191], while no clear findings emerge for gender (e.g. [49] versus [4]). Education has a

negative correlation with sharing, meaning higher educated people tend to share less data [298]. Experience as another antecedent indicates that positive historical experiences with a counterpart reinforce information disclosure [34, 311]. The same holds for personality traits, in particular trust and concern, which both impact the willingness to disclose data directly [34]. Additionally, Aivazpour and Rao [4] found that impulsivity has a positive effect on disclosure. To assess cultural differences, the five-dimensional framework of Hofstede [175] is applied by Krasnova et al. [214]; however, we do not go into detail here and refer to Wurmer et al. [407]. Lastly, people can be assigned to three different clusters that define their general attitude toward privacy (personal disposition) [399, 408], although there are studies to the contrary [211].

Situational Antecedents *Situation* of information disclosure is different for each decision but may show a common ground w.r.t. the factors type of information [228, 335, 387, 430], anonymization [40, 430], scope of use [191, 228], control [207, 228], retention period [228, 312], fairness [243], and trust [49, 153, 207] as well as the type of firm / website [228, 312]. People tend to share data that, at first glance, is not relevant to privacy. Bounded rationality as well as the benefits offered [387] undermine this concept. Anonymization is a significant driver of the willingness to share data with people willing to disclose information when it is not identifiable [40]. There is consensus that the ability to control or influence data processing is positively correlated with the intent to disclose [207, 228]. This is related to fairness when users can understand why certain information should be collected [243]. Trust towards the counterpart is a significant driver for the decision-making within the information disclosure process as it is omnipresent in every situation. While trust is a bidirectional concept between two entities, it can be extended to groups or types of firms and websites. Users show preferences for specific branches. For example, they prefer to share data with entertainment websites in favor of banking sites [228, 312]; however, this could be associated with the kind of data requested by the respective service [312].

Importance
for the
progression
of this work

This section provides essential information on privacy and the related information disclosure process. We will collect the knowledge when developing Privacy by Design-based architectures in Part II where situational antecedents and cognitive perception are relevant. Also, when analyzing privacy threats related to disclosure of sensor data in Part III, we reflect that benefits and costs are hard to grasp in specific situations, as mentioned previously. Understanding the information disclosure process and its complexity also drives the design of the Privacy Enhancing Technology (PET) presented in Part IV.



Research published in relation with this dissertation.

Figure 1.1

About this Work

1.3

This work tackles the challenging topic of dual-use of sensor data. Sensor data can be a reliable and comprehensive source of information but at the same time pose a severe threat to the interests of a person. The purpose of this work is to provide a comprehensive picture of what appears to be a dichotomy.

During the course of the research efforts, a total of 14 scientific articles have been developed and published³, all of which are related to the topic of this dissertation. The chronology of these publications is depicted in Figure 1.1. They use the methodologies described in Section 1.3.2 as a scientific basis to answer the research questions posed in Section 1.3.1. The approach presented in Section 1.3.3 is followed and accompanied throughout this dissertation to address the vast topic in an appropriately structured manner.

Research Questions

1.3.1

Seemingly unlimited availability of data offers an indisputable potential for novel use cases, especially in the mobile environment, as illustrated in Section 1.1. At the same time, their relevance makes them critical to the privacy and intentions of a user (c.f. Section 1.2) who, as an information dissemination entity, participates greatly in an information system. This tension raises the following research questions:

³ At the time of writing, Wurmer et al. [407] is currently under review

1. *How can the inertial sensor data of smartphones be collected and evaluated as robustly as possible in cars?*

High quality, high frequency, and accurate data are crucial for all use cases relying on sensor data. Modern applications such as Usage-Based Insurance forgo the use of dedicated hardware for economic and adaptability reasons and employ the ubiquitous smartphone for data acquisition. The research question will discuss to what extent data can be collected efficiently and robustly in different vehicles taking into account the heterogeneity of smartphone sensors and environments. Further, an exemplary implementation will be analyzed in-depth, assessing its quality.

2. *How can smartphone sensors be used as enablers for privacy-friendly applications in traffic scenarios? What possibilities do they open up?*

Multistakeholder environments require diversified consideration of individual interests and concerns. The research question aims at demonstrating the compatibility of conflicting security objectives such as availability, integrity, and confidentiality by means of concrete applications. In fact, it motivates a critical design process that identifies Privacy by Design as a driving factor.

3. *What privacy risks are induced or increased by the omnipresence of smartphones in vehicles?*

Exhaustive data gathering in the user domain results in a severe threat to privacy, further stressed by the overwhelmed user. This research question examines what information can be extracted from the data and to what extent insights can be drawn from individuals. The question is intended to make a significant contribution to eliciting critical awareness in the collection, distribution, and analysis of sensor data as part of various processes in an information system.

4. *What methods can be used to protect the privacy of users, especially in the Pay-How-You-Drive scenario?*

Building on the findings of the previous two research questions, this research question presents a method that protects a user's interests and implements them in a technical and verifiable manner. Special focus is placed on the diversity of users and their interests. In the course of answering this research question, both the functionality and the limitations of external protection methods for existing applications are considered.

Research Methodology

1.3.2

A methodology defines itself as a contextual framework to accomplish a given task using a set of specific instruments [402] to eventually support or achieve the generation of new knowledge applicable to the general public [375]. The European community of business informatics distinguishes two dominant approaches, namely design science and behavioral science [402]:

- ▶ **Design science** as the first paradigm strives to gain knowledge by creating and evaluating new and innovative solutions in the form of models, methods, or systems that are beneficial for applied usage.
- ▶ As the second paradigm, **behavioral science** is an observant science that tries to analyze the behavior and impact of existing information systems on business to generate new knowledge.

Motivated by the research questions and the goal of shedding light on the benefits and risks of sensor data, the design science methodology is preferable. In particular, we rely on the Design Science Research Methodology Process Model of Peffers et al. [292], outlined in Figure 1.2. The Design Science Research Methodology Process Model is a problem-centered approach that presents for each identified task an artifact that is successively expanded, scrutinized, and revised. It is organized into six activities that do not necessarily need to be completed in order. Depending on the level of existing knowledge throughout the community, different research entry points may be chosen. The fact of different depths of problem awareness is also present in the topic of dual-use of sensor data, which makes this approach appropriate.

Methodology according to Peffers et al.

We will now briefly introduce each activity of the Design Science Research Methodology Process Model (c.f. Figure 1.2).

Design science research methodology process model

Identify the problem & motivate The first step is to identify the problem and motivate the intended research to define the research boundaries. Depending on the size of the problem, it may be meaningful to atomize the problem so that an adequate solution can be provided that can grasp the complexity comprehensively. Furthermore, it is important that such a solution is acceptable and attracts awareness of the problem throughout the research community. Within this work, the boundaries were defined by the research questions defined in Section 1.3.1. Problem identification will be pursued by conducting literature reviews found at the beginning of each part, explicitly addressing each topic.

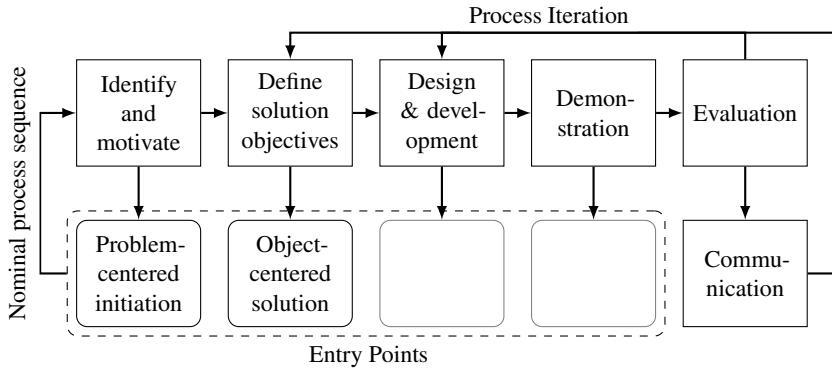


Figure 1.2 Design Science Research Methodology Process Model. The process model is composed of six interconnected activities following the problem identification and solution. Not all entry points are considered in this work. (based on Peffers et al. [292])

Define the objectives for a solution The second activity is a requirement analysis of the solution to be developed based on the defined problem. It is necessary to have a solid knowledge of state-of-the-art solutions to define a delta based on it and to determine the result specifications accordingly. This is done in the respective chapters based on studies (Chapter 9) and Structured Literature Reviews including related work (see Chapters 4, 7, 8 and 10 to 13).

Design and development The design and development phase is the third activity and aims to create artifacts according to the defined objective. In fact, this phase includes not only the construction of the artifact (development) but also its desired functionality and the architecture (design), with the research contribution being unambiguously recognizable. Within this work, each part presents one or multiple artifacts either in the shape of constructs (c.f. Chapter 9), models (c.f. Chapter 14), methods, or instantiations (c.f. Chapters 4 and 8) [171].

Demonstration The fourth phase is intended to present the feasibility of the artifacts to solve the problem stated in steps one and two. We conduct experiments (c.f. Chapters 4, 8, 11, 12 and 14) and simulations (c.f. Chapter 7) using the methods explained in the previous section. Each scientific environment is presented accordingly to enable clarity and replicability.

Evaluation The evaluation's purpose as the fifth step is to understand and show the ability of the artifact to solve the problem. This activity entails comparing the solution's aims to the actual results obtained through the usage of the

artifact. Qualifying suitability can be established, for example, by metrics and key performance indicators, but it can also be done through empirical proof. If needed, a loopback to previous steps, particularly to the third activity, can be done. In this work, mainly quantifiable measures are used, which are obtained with the help of the methods specified under the fourth activity. The metrics are presented in detail at the beginning of the respective evaluations within the chapters.

Communication Artifacts need to be presented in a consolidated and integrated way if they provide a novel and, in fact, feasible solution to a given problem in the sixth activity. Therefore, most of the work presented in this dissertation has been published (c.f. Figure 1.1), became open source (c.f. Chapter 4), and made available to the scientific community to foster insight into the relevance of the research, but also to encourage further activities based on open questions.

Due to the size and complexity of the problem that is the subject of this dissertation, the process is broken down accordingly [292]. One or multiple sub-artifacts emerge for each of the research questions and are related to each other so that they can provide an answer to the initial dichotomy in their entirety. The added value for the research community consists in the expansion of knowledge and the practically-oriented demonstration of the compatibility of seemingly contradictory requirements, primarily integrity and confidentiality, with privacy. Due to this requirement, appropriate research methods are selectively chosen, which allow a problem-oriented and pragmatic approach (e.g. formal-conceptual design, prototyping, laboratory/field experiments, or grounded theory) [402].

Research
methods

Structure of this Work

1.3.3

This dissertation aims to overview the possibilities and threats associated with smartphone sensor data. The work is intended to be comprehensive, addressing several facets of the subject holistically. As mentioned in Section 1.3.2, it is only feasible to answer the research questions given in Section 1.3.1 by deconstructing the complexity. This is achieved by separating the work into four parts, each of which discusses a different aspect of the subject and answers a specific research question, and is illustrated in Figure 1.3. Additionally, one can see the relations and implications that each part has on the other sections of this dissertation. We also feature short sidebars introducing related topics at appropriate points. These should be beneficial for the understandability of the currently discussed statements but are not intended to provide a thorough overview. We will now briefly introduce each part.

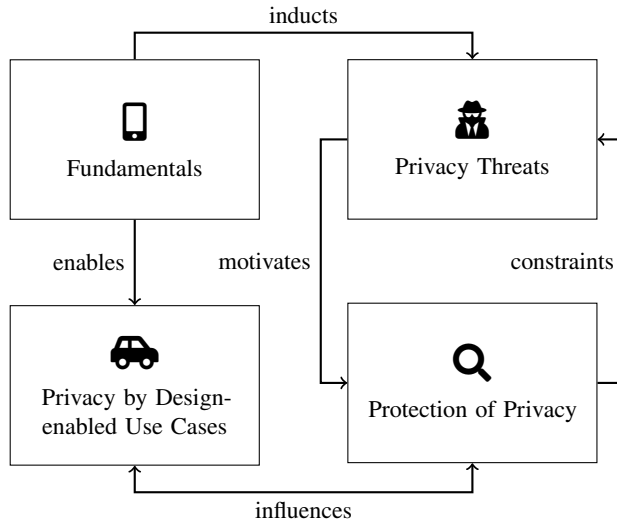


Figure 1.3 Overview of the topics which are the subject of the dissertation including their relations.

Chapter 1 first introduces the subject area and defines the concept of privacy as it is understood in the context of this dissertation.

Part I Fundamentals

The first part discusses the fundamentals to establish a common understanding of the sensor-driven mobile environment. Chapter 2 introduces the fundamentals of sensor data and connects them to the mobile environment. Additionally, idiosyncrasies are discussed that must be taken into account when sensor data is processed. Following that, we show in Chapter 4 an artifact that targets the first Research Question. This chapter discusses a method for collecting sensor data from a smartphone utilizing a range of factors. On the basis of these discoveries, Chapter 5 offers a real-world data set that serves as the foundation for the empirical evaluations in Chapters 8, 11, 12 and 14.

Part II Privacy by Design-enabled Use Cases

The next part discusses creating sensor-based applications that may be achieved using sensor technologies accessible on a mobile phone. Thus, it relates to Research Question #2. To provide a satisfactory response to the topic within the

confines of the dissertation, the fundamentals of privacy-friendly processing while retaining user interests are introduced and addressed (cf. Chapter 6). The term Privacy by Design is introduced. A Privacy by Design application demonstrating how to safeguard a user's movement pathways despite the transmission of descriptive data is motivated and detailed in Chapter 7. In Chapter 8, the informative potential of sensor data is extensively explored in a mobile setting. It is demonstrated how the balanced processing of sensor data may accommodate divergent requests from many stakeholders.

Part III Privacy Threats

Part three focuses on the abusive potential of privacy in accepted and applied business models. Therefore, this part focuses on Research Question #3. The business model in focus is the one of Usage-Based Insurance with its two manifestations Pay-As-You-Drive and Pay-How-You-Drive, all of which are introduced in Chapter 9. The relevance to privacy is explicitly mentioned and further stressed in an overview of side-channel attacks that specifically target sensor data as provided by smartphones in a mostly unprotected manner (c.f. Chapter 10). Based on the insight, two attack paths that focus on the UBI domain are developed and presented. Both show how data can be misused, which must be considered a threat to privacy. The first approach focuses on identifying people from pseudonymized driving histories, while the second attack traces the route driven without relying on location data. With the help of both attacks, a comprehensive picture of a person can be drawn.

Part IV Protection of Privacy

The fourth and final part attempts to use the findings of Part II to reduce or completely mitigate the threats of Part III (c.f. Research Question #4). To this end, a Structured Literature Review is first performed in Chapter 13 to analyze current building blocks suitable for this task. Building on the UBI environment introduced in Part III, we show that supposedly conflicting applications and interests can also be reconciled using the Privacy by Design paradigm and related constructs.

Chapter 15 again summarizes the findings of this work and situates them within the research questions from Section 1.3.1.

Part I

Fundamentals



Sensors are used to capture their environment's physical, chemical, or material properties or to detect changes within their observation. This information can be qualitative or quantitative, the latter typically being measured quantities. The properties are detected through physical, chemical, or biological effects and transmitted in the form of electrical pulses. For example, sensors react to light, heat, noise, pressure, magnetism, or movement. Active or passive sensors can be found. Sensors of the former class generate an electrical signal to manipulate the environment and measure the corresponding effect, whereas passive sensors are actuated by the measured variable and do not require any auxiliary power. Typical examples of the former are accelerometers that exploit the piezoelectric effect or laser sensors for ambient measurements. Hygrometers or magnetometers, for example, belong to the class of passive sensors.

Introduction

The counterpart of a sensor is an actuator, both of which work together in the sense of *measure and react*. The output information of a sensor can serve as the input variable of an actuator, which is a corresponding control element in a control loop. It translates the electrical input signal of the sensor into a (mechanical) effect, for example, the opening of a valve.

Actuators as a companion

There is a distinction between analog and digital sensors. Analog sensors continuously measure an effect, outputting any value over the measurement range, eventually yielding an indefinite number of output states. Digital sensors, on the other hand, emit only certain states and, according to their designation, transmit the data digitally. A special kind of digital sensor is a binary sensor that transmits only two states, indicating the presence or absence of a defined situation. The conversion of the measured signal is done directly in the sensor into a digital representation. This minimizes disadvantages such as the potential change of the analog signal during transmission and increases the integrity of the measurement (for example, by error compensation in the device), but possibly at the expense of accuracy since only discrete states can be represented. The difference in the output variable is illustrated in Figure 2.1. The analog sensor continuously describes an effect as shown in Figure 2.1a, while the digital sensor discretely describes it, with a possible loss of information, recognizable in Figure 2.1b.

Analog and digital sensors



a) Continuous signal. A continuous signal is sampled with no discrete time interval.

b) Discrete signal. A discrete signal samples the underlying effect at specific time points.

Figure 2.1 Example of the same event recorded continuously and discretely. The same event yields different shapes depending on the processing of the signal.

Time
interval and
frequency

From the graph, it can be concluded that the discrete measurements of a digital sensor occur at specific points in time. These time points are denoted by t in the context of this work, where $\Delta\mathcal{T}$ describes the (absolute) time interval between two measurements t_1 and t_2 . The smaller the time interval, the more accurately the underlying effect to be measured can be captured. When describing the intended time interval $\Delta\mathcal{T}$, we refer to the frequency of a sensor, which is defined as $f = \frac{1}{\Delta\mathcal{T}}$.

In the following, the sensors relevant to this work are presented. Subsequently, example use-cases are introduced that focus on location-based context. The chapter concludes with a discussion of the quality of the sensors and opens the topic of error causes and handling in connection with the processing of sensor data.

2.1 Overview of Sensors and Sensor Data

Sensor data and their elaboration are the main subjects of this work. Therefore, sensors are presented, in general, in the following. Due to the emphasis on mobility, a comparison between vehicle sensors and sensors such as those found in smartphones becomes apparent at this particular point. This section concludes with an introduction to sensors that are widely used and vital to the further progression of the dissertation.

Typical sensor application fields include, but are not limited to, health, industry, entertainment, mobility, and Internet of Things. For instance, the industry employs sensors to reduce the downtime of their machines by collecting various types of metrics and characteristics of machines to proactively plan the mainte-

nance window accordingly in a dynamic and predictive manner. This process is called predictive maintenance and is enabled by extensive sensor technology in machines and the entire production chain.

Sensors in General

2.1.1

Sensors have been found in conventional vehicles for quite some time. However, sensors and their capabilities are likely to be known to the general public from the smartphone domain. Although both domains are considerably different, for example, in their requirements, the informative value of the sensors for certain conditions in the context of mobility is comparable.

Vehicles

Vehicles are equipped with a multitude of sensors, although the number of sensors has increased dramatically in recent times. Within five years, the market for sensors in vehicles is expected to grow by more than 60 % in revenue [368]. This is due to the need for modern vehicles that have Advanced Driver Assistance Systems (ADAS) onboard. These systems support the driver or take over tasks from him. For example, cruise control helps maintain the distance from the vehicle in front and performs braking and acceleration maneuvers independently. A lane departure warning system can also help the driver avoid drifting out of his lane. Thus, Advanced Driver Assistance Systems (ADAS) contribute to increasing traffic safety.

Increasing spread of sensors

Vehicle sensor data can be accessed by external systems using an On-Board Diagnostic (OBD) adapter. OBD is a vehicle diagnostic method that monitors the systems, especially the ones that affect exhaust emissions, in real-time while the vehicle is in operation. However, it also allows access to other control units and to query various sensor values as they are transmitted to the control unit by the sensor. It is thus possible, for example, to access ABS sensors or engine speed in real-time, but also data such as acceleration and rotation data. Acceleration and rotation data are of particular interest in the context of this work. Data can be taken with the OBD connector. Communication between different Electronic Control Units (ECUs) in the vehicle is typically done via a bus system called Controller Area Network (CAN). This is a message-based system in the sense of the multi-master principle, i.e. it connects several ECUs that can communicate at the same level. Higher-priority messages are favored in the event of a message collision. It was first developed by Robert Bosch GmbH in 1983 and has since established itself on the market. CAN is internationally standardized in ISO 11898-1.

Accessing vehicle sensors

Drawbacks
of vehicle
data

Although data accessed through the CAN bus is of high quality as it is directly derived from a vehicle and thus allows to gain information about e.g. its movement, two major drawbacks are making it uninteresting in the context of this work. First, dedicated hardware, such as an OBD dongle, is needed to access the data along with specialized software capable of understanding the CAN bus standard. Second, physical access to a vehicle is required to install the hardware. Together, both drawbacks prevent scalability, cost efficiency, and retrofitability.

Smartphones

Accessible
sensor data
interfaces

Smartphones (also mobile devices in general) are likewise equipped with sensor capacities (c.f. Figure 2.2), but, in contrast, allow data access to onboard sensors using well-documented Application Programming Interfaces (APIs). Any modern smartphone is equipped with sensors to allow, for example, to flip the display depending on the device's orientation, measure if a person is moving, or adapt the display brightness automatically. Application developers can easily implement their logic by accessing sensors using a standardized interface.

Varying
quality

However, depending on the sensor and task, the data is of inferior quality compared to the data directly accessible through the CAN bus [198]. Imagine a smartphone placed in a vehicle. The vehicle's velocity can be accessed in sound quality using the wheel sensors' data when directly tapping from the car's internals. However, when trying to derive the velocity using the smartphone, one has to use the accelerometer subject to different disruptors (see Chapter 4). Other literature confirms that sensors can still produce readings close to those of a vehicle [131]. Additionally, the sensor quality varies between devices [219].

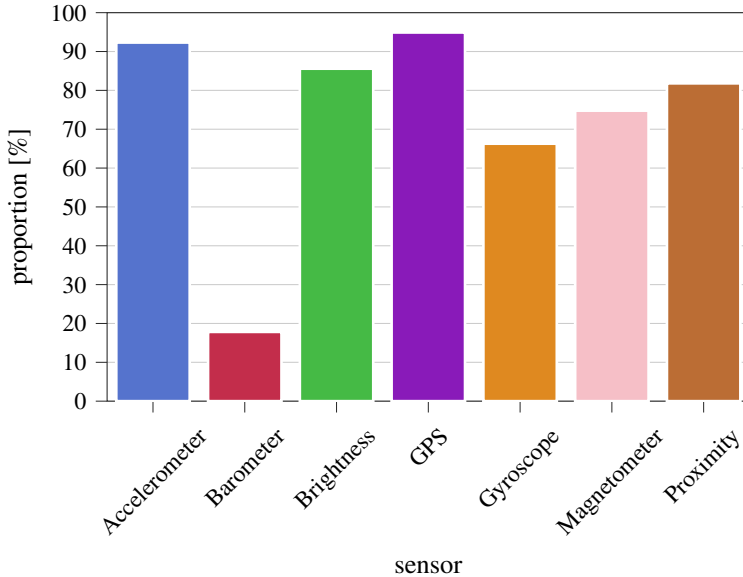
Limited
sensor
information

Furthermore, even though smartphones are equipped with sensors, there are limited specifications measurable by smartphones in the context of vehicles compared to direct access to sensors onboard a vehicle [151]. The example mentioned also illustrates this fact. When using the smartphone, it is sometimes necessary to derive or extrapolate data to make statements similar to those that would be possible using OBD, eventually introducing another factor of uncertainty.

Sidebar A Floating Car Data and Floating Phone Data

Data generated by a vehicle is commonly called Floating Car Data (FCD). The

¹https://geizhals.de/?cat=umtsover&xf=3229_2020; accessed September 9, 2021



Share of sensors build-in a smartphone released in and after 2020 after a market analysis. In total 1352 devices are listed. Shown are sensors with a significant market share. Accelerometer and GPS are available in nine out of ten devices. (data by Geizhals.de¹)

Figure 2.2

principle of FCD is to collect real-time traffic data using vehicle sensors, locating the vehicle via Global Positioning System (GPS), mobile phones, or using Intelligent Transport System (ITS) infrastructure (see Section 7.2) over the road network. Data submitted to a centralized processing entity can vary but typically include speed, location, and direction.

Depending on the information collected and sent, the term Extended Floating Car Data is also found in the literature. It is commonly used when data is generated by ECUs and further processed in a vehicle. It allows one to craft messages that include information about e.g. the condition of the road or traffic flow, as well as situational traffic disruptions. However, the definitions alternate [53].

Transferring the concept of FCD to the area of mobile devices, particularly smartphones, formed the term Floating Phone Data (FPD).

2.1.2 Types of Sensors

We now briefly describe sensors commonly found in a smartphone w.r.t. this work, mainly following the specifications of Android on sensors [98]. Three different categories of sensors are available in current mobile devices:

1. **Motion sensors** measure acceleration and rotational forces in a three-dimensional space.
2. Environmental parameters such as the air temperature are recorded by **environmental sensors**.
3. The physical position of a device can be determined using **position sensors** which also includes GPS or the magnetometer.

Distribution
of sensor
types

A market analysis illustrates what sensors are built into current devices. Figure 2.2 shows the results. Because this work focuses on location and mobility, we especially emphasize a basic set of sensors for this work. This leaves out sensors such as barometer, proximity, and brightness, although they have been used in this academia [172, 341].

Coordinate
system

As will be seen, this work uses data from sensors of the three categories mentioned above (see Chapter 5). Data is measured with an Inertial Measurement Unit (IMU), integrating the gyroscope, magnetometer, and accelerometer into a single device. Each sensor represents a motion sensor and measures a 3-dimensional vector that represents changes at a given time. The coordinate system considered for the vehicle is shown in Figure 2.3. The lateral (also transverse or pitch) axis is called x , the longitudinal (or roll) axis is defined as y , and z represents the normal (or yaw) axis. A smartphone also has a coordinate system similar to the vehicles'. However, they do not necessarily need to be aligned and can have any orientation to each other, but must remain stationary. We focus on the challenge of aligning both independent coordinate systems, a problem called *smartphone-vehicle alignment*, in Chapter 4. Due to the focus on smartphones in this work and their respective coordinate orientation, it should be noted that the coordinate system of a vehicle is purposely different from the one defined in DIN 70000. Furthermore, the direction of the axis, i.e. the output of negative or positive values, can vary between devices. However, we describe the sensors using the right-hand convention as shown in Figure 2.2. This is also the orientation found in most cases.

Gyroscope

First, a **gyroscope** (readings denoted as gyr) is a motion sensor that measures the rotation of the device around three axes in rad s^{-1} using the local coordinate

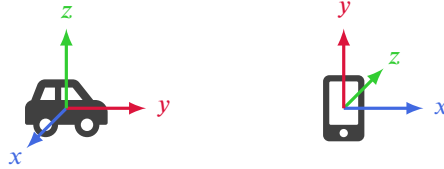


Figure 2.3

Overview of the coordinate system of vehicles and smartphones. To offer comparable sensor readings across different device orientations, the coordinate system of a smartphone should be aligned with the reference coordinate system of a vehicle. Rotation vectors can be used to obtain the alignment within a process called smartphone-to-vehicle alignment. (based on Project [302])

system. Thus, the gyroscope provides information on the angular velocity around the said axes [385]. The measurement sign denotes the direction of the rotation, with positive readings indicating a movement clockwise, while negative values represent a movement counterclockwise. For example, if the mobile device rotates to the left, readings from the gyroscope along the z -axis will yield positive values, and rotation to the right will result in negative readings. Consequently, this sensor is suitable for detecting changes in driving directions.

Next, a **magnetometer** (readings denoted as *mag*) is able to sense the magnetic field in μT , which acts on the device on the three axes [386]. Since Earth's magnetic field is also organized in a three-dimensional space, one can use this position sensor's readings in x and y axis to determine a compass heading [187]. This allows one to draw conclusions about a vehicle's heading, although the knowledge is only microscopic. However, the magnetometer is subject to disturbances caused by other magnetic fields or surrounding materials since its vector always points to the strongest magnetic field. The literature came up with approaches to correct readings in the latter case to improve the accuracy of the detection of the magnetic field of Earth [187].

Magnetometer

An **accelerometer** (readings denoted as *acc*) is used to detect the current acceleration in m s^{-2} as experienced by a mobile device. Furthermore, the motion sensor measures the acceleration, i.e. the change in velocity of a device w.r.t. time along the three axes x, y, z [384]. Measurements of the accelerometer also include gravitational force. When cleared from this disturbance, the accelerometer gives insight into a vehicle's movement state by analyzing the y -axis: acceleration yields positive values, while braking results in negative values. There are virtual sensors available, e.g. in Android, that rely on the accelerometer, such as the

Accelerometer

gravity vector (output acceleration due to gravity on all three axes) or the linear acceleration (tries to remove gravity from accelerometer readings).

Position Pars pro toto, GPS represents a **position sensor**, which allows the position of a device to be determined unambiguously in space. This usually requires a line of sight to a Global Navigation Satellite System (GNSS), whose satellites continuously transmit their position and time. A GNSS receiver, such as a smartphone, is able to determine its position and altitude based on at least three satellite signals and their associated information. The position sensor requires relatively much energy compared to Inertial Measurement Unit (IMU) sensors [44, 414]. This limits the use in a mobile environment characterized by energy efficiency.

Sensor representation in android All said low-level sensors are available at least since Android 2.3 (API level 9). Physical sensors are represented by base sensors available for uses that may apply preprocessing steps such as temperature correction, scale calibration, or bias compensation [302]. Thus, a sensor is a virtual representation of the underlying hardware sensor. Multiple new sensors can be created using the data from these above sensors in different ways, called virtual sensors in Android. Additional sensors such as the gravity sensor, linear acceleration sensor, or rotation vector sensor are subject to Chapter 4. They can be accessed using the Sensor Event API on Android.

2.2 Location, Context Awareness and Motion Sensors

This section contains a definition of location in the context of computer science, as this term may be understood differently. It contains results derived from reviews of the literature of Teh et al. [367] and Wasserle [S16]. Particular attention is paid to mobile use cases, eventually relying on sensor data from smartphones with a clear focus on location.

Exemplary use cases Motion sensors or sensors, in general, can be applied to solve different problems, as already mentioned. For example, sensors can be found in various environments, not only in vehicular environments but also in industrial processes to monitor the state of machines, in smart cities to analyze and steer specific properties, or in health care to make the lives of individuals safer and more convenient. Within the context of this work, motion sensors are of particular interest that have been introduced in the previous section. The mentioned use cases may depend on the location of an entity (including humans). However, location can be

related to motion sensors that are used to sense the current environment, setting, or context.

Consequently, we need to define the term location itself as it has different meanings depending on the context. It can be understood as a single point uniquely identifying a position or place, but it is also used to describe a journey i.e. as a list of successive positions. In this work, a single position is considered a *location* with a list of locations called *trajectory*. The location can further be divided into an absolute or a relative position as well as co-location. Absolute location is the unambiguously defined position in space, as determined by e.g. GPS. The relative location is the location w.r.t. another object. Finally, co-location may not indicate an absolute location but means a location shared by more than one entity.

Definition of location

The location can either be defined using various methods, the GNSS being the most prominent. A concept of deriving context from the current situation of entities is called “*context awareness*”; hence, this concept eventually includes the location [104]. Context-awareness can be considered a foundation for crowd-sensing, as it enables a user device, such as a smartphone, to detect, interpret, and react to changes in the environment. It is an essential building block for Weiser’s vision of ubiquitous computing. According to Dey [104], the context is no longer limited to location but considers a process with user involvement (active or passive). Context-aware systems can adapt to a situation and provide i.a. convenience to users by automatically sensing the current position, such as a meeting room, and include this information in a decision-making process, such as turning off any notifications not to disturb the user; hence triggering activities based on context. Also, a mobile device can detect if the current context is driving a vehicle and can start related applications, such as navigation, thus using a sensor to perceive a setting. In summary, Schmidt et al. [333] defines multiple benefits of context-aware systems such as adapting interfaces, focusing information presentation, relieving a user from interaction, or building smart environments. Consequently, mobile sensors, including motion sensors, are needed to build ubiquitously context-aware applications. Context is an essential companion to location, as the more contextual information is provided, the better the user or the system experience [150].

Context awareness

This motivates the need for robust and pervasive detection of the location. As the position sensor in mobile devices requires much more energy than other sensors, other approaches are favorable for detecting the current position or environment. Therefore, we want to emphasize the usage of smartphone sensors for location, indoors or outdoors. At the same time, this can be helpful but

Location

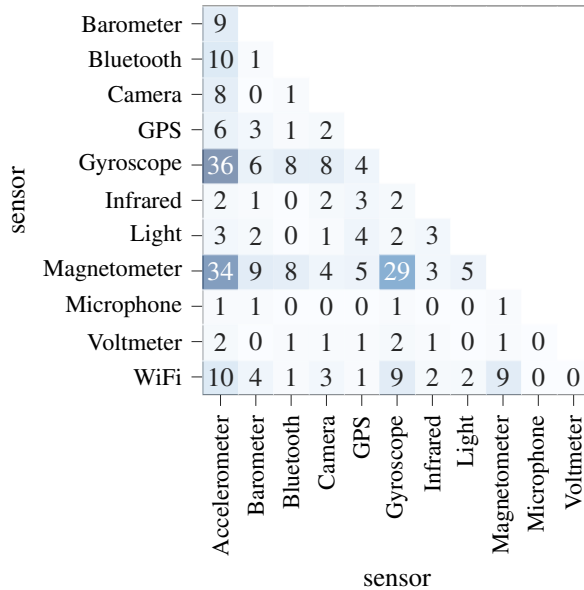


Figure 2.4 Heatmap illustrating sensor combinations found in literature for the achieving user localization in indoor and outdoor scenarios. The accelerometer, gyroscope, and magnetometer are often combined and the dominant combination in terms of tracking. (based on Grohmann [S5])

also dangerous, as Section 1.2 and Part III will show. As part of Grohmann [S5], a Structured Literature Review (SLR) was conducted that identified 49 articles that used sensor data to determine the whereabouts of a person. Papers that used GPS as a basis were excluded. The combination of accelerometer, gyroscope and magnetometer was found to be dominant in different settings (c.f. Figure 2.4). Thus, of 49 works, the accelerometer is combined 34 times with the magnetometer and 36 times with the gyroscope. With nine out of sixteen observations, this is the primary choice when three sensors are combined.

Dominating approaches

Methodically, Pedestrian Dead Reckoning is the most common approach in the literature. In this method, the relative position determination takes place, where position changes can be iterated to a target based on an initial position. A position change can be determined or estimated in various ways. For example, a person's steps can be counted in the context of Human Activity Recognition or, as done in this work, based on curve trajectories and distance paths in the mobile street

scenario (see Chapter 12). Path-loss models, in which the signal strength of a radio connection is evaluated over time, are also used to determine the position. Other methods include fingerprinting, camera-based localization, and ambient noise.

Quality of Sensor Data

2.3

Sensor-based applications commonly rely on accurate data. Although smartphones offer the ability to gather data at a low cost, their accuracy and correctness are still expected to be feasible in specific applications.

Kuhlmann et al. [219] systematically analyzed the accuracy of the sensor in the example of spatial orientation, i.e. measurements of a gyroscope. The authors used two leveled mounting devices to assess the sensor quality of 56 different smartphones running iOS, Android, and Windows Phone 10. The phones were tilted on the mounting device in specific orientations, while measurements were recorded on the smartphone using a dedicated app or a browser-based app API.

Literature review assessing gyroscope accuracy

Devices differ in accuracy, as the results illustrate. Some show mean deviations close to 0° and slight variances, while other devices pose mean inaccuracies of up to 2° , on some occasions reaching over 6° compared to the objective tilt. The deviations are higher for roll measurements than they are for pitch measurements. The software implementation does have an impact on the results, but only a minor one and can be neglected depending on the situation. Interestingly, even the same device models were not consistent across tests, although the results for a device are moderately consistent, which means that a deviation in pitch in one direction shows a positive correlation with the deviation in pitch at other angles. Different operating systems also had an influence.

Wide-spreading inaccuracies

The fluctuations in the output of the measured values by the sensors, therefore, differ depending on the device and the Operating System (OS). Thus, sensor data is subject to uncertainty [245]. Depending on the application, they can still be used if the deviations are taken into account accordingly. Usually, deviations are considered disturbance variables, and an attempt is made to minimize them. Minimization can be done on the software side or by hardware calibration. The latter is typically performed during production and stored in the device. It has also been shown that calibration must be performed separately for each device since it is device-specific. Zhang et al. [423] have shown that calibration can serve as a unique identifier for devices since the underlying entropy is sufficiently high (c.f. Chapter 10).

Minimizing fluctuations

2.3.1 Errors

Khaleghi et al. [206] and Teh et al. [367] describes known challenges that occur in the field of data processing. We briefly summarize them while focusing on the mobile device sensor context. The given errors significantly influence the integrity of measurements and eventually lead to wrong conclusions.

Outliers are values that exceed a specific range of expected values according to thresholds or specific models. They may also deviate a lot from previous and subsequent sensor readings.

Missing data or incomplete data describe observations that are not available for further processing. This holds for entire measurements that are absent or only specific dimensions that are not recorded (e.g. timestamp). There may be various sources for the origin, not limited to unstable sensor devices, faulty persistence, or any other kind of data manipulation.

Bias is also known as an offset and describes “*a value that is shifted in comparison with the normal behavior of a sensor*” [305]. The reasons are uncalibrated or low-quality sensors.

Drift are measured values that gradually move away from their true value over time, either constant or variable. The source may be a defective sensor.

Noise impacts the sensor and consequently its readings, which are not related to the measured object but have their course in e.g. environmental influences such as gravity. Noise may vary over time and depend on the environment.

Stuck-at-zero or **Constant Value** yield the same sensor readings over a period of time; either it may be zero or any other constant value that does not change no matter what influences act on the sensor.

Uncertainty is a state in which it may not be obvious if a sensor reading is correct or wrong.

2.3.2 Detection and Correction Methods

In addition to the challenges of processing sensor data, Teh et al. [367] presents different approaches to handle the limited quality of sensor data from mobile devices. It is crucial to first detect errors in the stream of sensor readings before adequately handling them. Teh et al. [367] list several methods for detection,

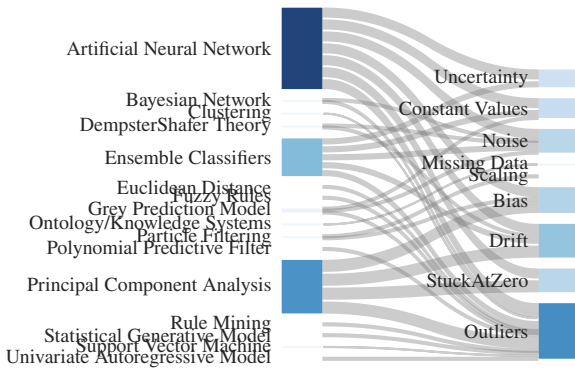


Figure 2.5

Methods applied to detect or quantify sensor errors organized by their frequency used. It is clear that Artificial Neural Networks along with Principal Component Analysis and Ensemble Classifiers are the most common ones, especially to detect errors like drift, outliers and noise (based on Teh et al. [367]).

quantification, and correcting sensor errors in their survey. Their statements are based on 57 publications that have been examined.

Figure 2.5 depicts methods (right) for the detection and quantification of the errors presented (left). It is apparent that Artificial Neural Network along with Principal Component Analysis and Ensemble Classifiers are used for multiple errors and are preferred throughout the literature. Outliers are the error category that seems to be the most addressed. In general, approaches based on Machine Learning (ML) are dominant in the context of detection, quantification, and correction.

Error detection and quantification

To be able to evaluate the sensor data despite detected and quantified errors, it is necessary to correct or mitigate errors accordingly. Suggestions for this can also be found in Teh et al. [367]. Although all the described errors influence the resulting data processing steps, only a few papers addressed the steps of trying to correct the data. Papers addressing to correct data seem only to target missing data imputation, i.e. trying to estimate sensor measurement values for either missing data or removing noise. The corresponding papers use well-known methods such as Association Rule Mining, clustering, or k-Nearest Neighbor for this purpose. Association Rule Mining is the most common one and describes an approach where the most recent sensor values contribute to the sensor stream with higher contribution and thus may hint information for missing data. This

Error correction

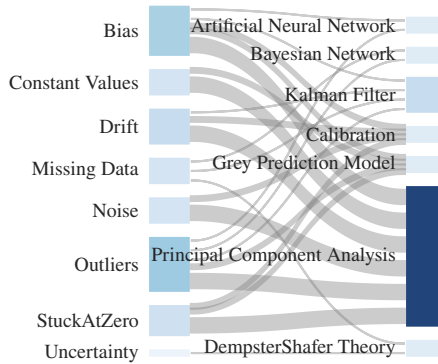


Figure 2.6 Methods applied to simultaneously detect and correct errors in sensor data streams. Identified approaches exhibit an overlap with methods for the detached detection of errors. Some approaches require additional knowledge or predeceesio steps. The most common error targeted are outliers. (based on Teh et al. [367])

claim may only apply to data without harsh changes in momentum. It may not be feasible for approaches like Chapter 4.

Combina-
tions of
detection
and
correction

However, in contrast, some papers combined error detection and correction in one step (c.f. Figure 2.6). The errors addressed were more extensive and covered all known defects. The basic idea is to use a behavioral model in which atypical measured values are marked as anomalies. They are replaced accordingly with an estimated value corresponding to the model. Similar procedures are used as in the case of detection, namely Artificial Neural Network (e.g. [182]), Bayesian Network (e.g. [97]), and Grey Prediction Model (e.g. [77]). In the context of mobile sensor data from an IMU, ML-based models may remove forces that produce acceleration or movement of a vehicle since these values are sometimes harsh changes in momentum. Furthermore, Teh et al. [367] identified Calibration-based methods as a viable approach to correct multiple defects at once. Also, since an environment evaluation has to be carried out at first [418] to gain knowledge about the defects of the specific device, and we already know from Kuhlmann et al. [219] that each device slightly differs in accuracy, such an approach will not scale well. Next, Kalman filter-based approaches, as presented by Feng et al. [128] are able to remove short-duration errors such as outliers, although they must be combined with other approaches to target long-duration errors such as drifts.

The articles present an approach to a specific research topic in the field of sensor data, but not inevitably in a mobile context. The approaches applicable in this environment must be checked. In principle, the movement of a vehicle alone causes significant changes in the force conditions in very short time intervals. For example, a short acceleration maneuver that involves a turn can fundamentally deviate from previous sensor data, which can pose a challenge to model-based methods.

Remarks

Privacy and Security Aspects of Sensor Data 3

This chapter introduces the present security concept of sensors in mobile Operating Systems (OSs), namely Android and iOS. It also outlines threats concerning privacy and security. In the example of the Joker/Bread attack, it is shown how attackers circumvent current security mechanisms in mobile Operating Systems and that there is further need for research and action.

The security concepts of Android and iOS distinguish between protected and unprotected data (including sensors) and activities. The category of protected elements typically includes GPS, camera, microphone, SMS, and contacts. To gain access, the app must request appropriate permissions, which must be considered by the user. This does not apply to the second class, the unprotected elements. These can be accessed, evaluated, or edited by an application at any time in the process. Low-level or base sensors are generally assigned to this category, including IMU. Hence, they are also called *zero-permissions sensors*¹.

Protected and unprotected sensors

This chapter first introduces the Android permission system before discussing its limitations and misconceptions. An example of how the permission system may be bypassed is presented afterward.

Structure

Permission Workflow

3.1

Current mobile Operating Systems implement a permission system to protect access to sensitive data and sensors. We explain the permission system at the example of Android [100]. Android is developed by the Open Handset Alliance as open-source based on Linux. It is the most used mobile OS. A comparable permission system is also present in iOS. Android uses a sandbox-based approach to isolate Java-based applications within the system; hence, each application has its own user ID. Resources such as data storage or sensors are

¹Starting from Safari 13, Apple protects motion sensors with runtime permissions. Also, their Core Motion framework implements security mechanisms for some motion events based on zero-permissions sensors.

not accessible (across processes) except explicitly requested and permitted. An extensive analysis of Android's permission system can be found in Almomani and Khayer [13].

3.1.1 Permissions

There are different types of permissions present in Android, namely install-time permissions, runtime permissions, and special permissions². They differ in the scope of restricted data given access to and the scope of restricted actions that an application can perform [100].

Install-time permissions

First, **install-time permissions** are given to an application once the user installs it from e.g. an app store after first presenting them to a user. Basically, this set of authorizations includes powers that change the state of a system only to a small or no extent. Furthermore, access to protected data is limited. An example that can be mentioned is querying whether the device is connected to a network (`ACCESS_NETWORK_STATE`). Next, **normal permissions** show little risk to privacy and system security but allow access to resources outside the sandbox. `RECEIVE_BOOT_COMPLETED`, for example, allows an application to wait until the system has finished booting.

Runtime permissions

In contrast, **runtime permissions** are also called dangerous permissions due to their ability to give access to sensitive and private data or allow performing restricted actions. They are requested at runtime, with the user needing to allow them explicitly. Of course, the request can be denied, or permissions can be revoked at any time. Additionally, Android 11 (API level 30) introduced one-time permissions as an advanced protection mechanism for particularly dangerous permissions such as location, microphone, or camera. Examples of one runtime permissions are `READ_SMS` or `ACCESS_FINE_LOCATION` being a one-time permission.

Special permissions

Special permissions are another dangerous and extensive set of permissions. Common apps cannot request them. In fact, they are reserved for the system itself or OEMs. For example, (`READ_LOGS`) allows access to system logs that may contain sensitive data. Lastly, **signature permissions** reflect the set of permissions that another application signed with the same certificate has already requested, and thus they are implicitly given to the application during installation.

² See <https://developer.android.com/reference/android/Manifest.permission> for a list of all permissions available in Android at the time of writing


```
1 <manifest ...>
2   <uses-permission
3     ↪ android:name="android.permission.ACCESS_FINE_LOCATION"/>
4   <application ...>
5     ...
6   </application>
</manifest>
```

Example of the `AndroidManifest.xml` (truncated). Permissions are requested by defining them using the `uses-permission` tag that includes a unique permission identifier. In the example, the requesting application wants to access the precise location of the device.

Listing 3.1

In total, there are 167 different permissions present in Android 11 (API level 30), with 30 being dangerous and 47 normal ones [13]. In conclusion, the Android permission system is a fine-grained permission system enabling end-users to control actions and data available to the installed application, ultimately allowing a user to control his privacy. The system then enforces protective security mechanisms accordingly. However, the system is flawed, as we will discuss in Section 3.2.

Summary

Declaring

3.1.2

Android permissions are declared statically. They are bundled to a specific application version. The declaration of permissions is done in the app manifest. The manifest called `AndroidManifest.xml` contains information about an application, such as the name, starting activity, or requested permissions.

Static permission declaration

Listing 3.1 is an excerpt of the `AndroidManifest.xml` for some application. The application requests access to the fine location, which allows it to precisely locate a user using Android location services. Eventually, requesting `ACCESS_FINE_LOCATION`, gives an application access to the GPS sensor. Furthermore, an application can also declare whether access to this specific hardware component is essential or not [99]. If an application tries to access resources or execute an activity that is not listed in `AndroidManifest.xml`, the application ends up in failure.

Example

Sometimes permissions are not needed, for example, `READ_EXTERNAL_STORAGE` which allows an app to access data outside of its sandbox. Mostly, it is sufficient to store application-specific data within the sandbox. The developer's responsi-

Selection of permissions

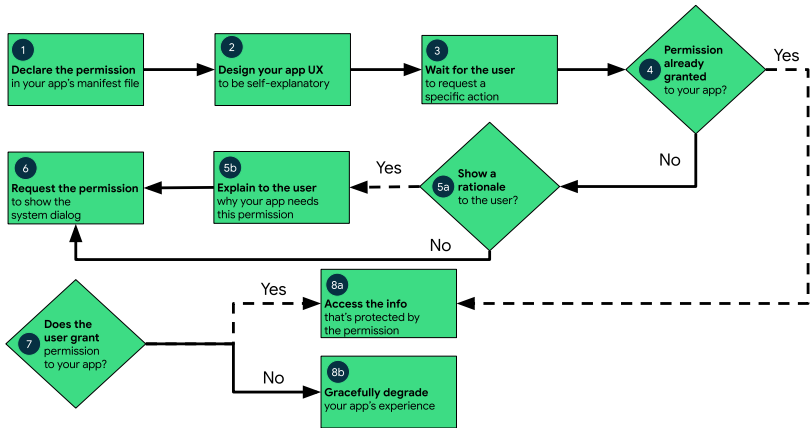


Figure 3.1 Overview of permissions requesting workflow. The process is separated into eight steps and provides general information about how to request permissions from a user from a developer perspective. [I3]

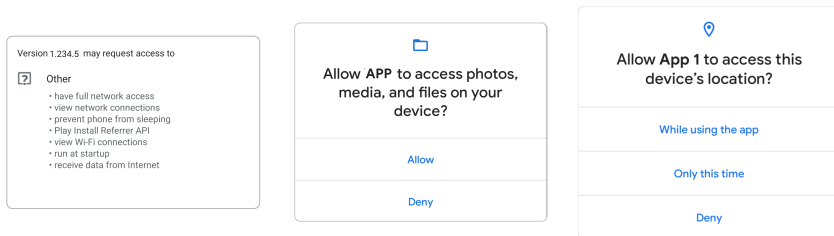
bility is to find a minimal set of permissions; apart from any drawbacks, this is considered best practice [100].

3.1.3 Requesting

Once all permissions are declared, the next step is to request them from a user. As explained, there is a difference between install-time permissions and runtime permissions. Since the former is granted during installation implicitly (as shown in Figure 3.2a), this section focuses on the latter.

Multi-step
workflow

Android implements the workflow shown in Figure 3.1. The request workflow must ensure that the user understands why the application needs specific permissions e.g. by showing an explanation. Thus, the request dialog, according to best practices, shall be shown once the application requires these for further processing [101] (step 3 & step 5). As one can see, the permission request dialog is a system dialog that cannot be altered by an application (step 6). The user can either allow or deny a request (c.f. Figure 3.2b). The app should respond appropriately (steps 7-8), i.e. “gracefully degrade (...) so that the user can continue using your app, possibly by disabling the feature that requires the permission” [101]. This is a one-time process since already granted permissions do not need to be re-requested.



a) **Install time permissions.** [I1]. b) **Runtime prompt.** [I4]. c) **One Time Prompt.** [I2].

Overview of the three different kinds of permission request screens. The Operating System selects the correct dialogue based on the information provided by a developer in combination with the permission type.

Figure 3.2

This fact has changed with Android 11 (API level 30), which introduced one-time permissions [274]. This permission aims to protect particularly sensitive sensors, namely location (e.g. GPS), microphone, and camera. They are not persistently given to an app and have to be re-granted. The app can request access to the data only for a period of time and depending on the user's choice (see Figure 3.2c). If a user grants access to the sensor "while using the app", the sensor and related data can be used infinitely while the app is in the foreground, but only a short time in the background (unless explicitly granted using the Android system settings). A user can restrict access to the sensor even more by granting the permission "only this time". In this case step 4 in the request workflow (c.f. Figure 3.1) always returns no. Finally, a user can, of course, deny a one-time request similar to other runtime permissions. As a side note, Android 12 (API level 31) will further allow a user to alter permissions within that prompt, in particular location ("approximate location" by downgrading `ACCESS_FINE_LOCATION` to `ACCESS_COARSE_LOCATION`).

One-time permissions

To further protect a user's privacy, sensitive runtime permissions of unused apps are auto-reset by the system starting from Android 11 (API level 30) periodically. However, a user can disable this automatic behavior for a specific app. In addition, Android separates between foreground and background processes and enforces different policies depending on the current type. For example, background processes are limited in accessing sensors.

Housekeeping of permissions

3.2 Discussion on Permissions

Android employs several security mechanisms to ease the burden on the user and protect his interest. The protection of privacy in Android is based on two pillars. First, permissions are predefined by the Open Handset Alliance and resources are segregated accordingly (*organizational-based privacy*). Users can make a personal decision using these permissions to determine to what extent they implement their claim to privacy (*user-based privacy*).

3.2.1 Organizational-based Privacy

Constant
adaption of
security

In the beginning, there must be an understanding of the potentials and threats that can emanate from sensors and other resources. Then an authorization system must be established that has probate means to protect these elements accordingly (i.e. capable of providing and enforcing security). Android has a system as previously described [13]. However, the security system of Android is in constant change. This can be trivially deduced from quantitative figures: During its development, the number of available permissions has increased from 73 (Android 1.x API level 1-2) to 167 (Android 11 API level 30). This corresponds to more than a doubling. Nevertheless, there are currently only two or three critical levels in terms of permissions, normal and dangerous (see Section 3.1.1 and Almomani and Khayer [13]). However, the assessment of which category a permission belongs to is made by the system design but has a direct impact on the user.

Developer-
driven
explanation

Android, however, shifts the task of explaining permissions to a developer. In 2012, Felt et al. [127] already claimed that the permissions shown to users are resource-centric. As seen in Figure 3.2b this is still the case in the current Android version. Android describes in its best practices that a developer should show “an educational user interface to the user” and that this interface should explain “why the feature [...] needs a particular permission”³. It is not clear how this practice is applied in the wild and if it is somehow validated. People also tend not to read these descriptions thoroughly [215].

Managing
permissions

The management of permissions, though, is a crucial task. The Android system provides sophisticated tools for it, such as the `AndroidManifest.xml` to define permission sets, with users to actively manage their privacy via the request dialogue or the ability to revoke permissions later. However, Kreuter et al. [215]

³ <https://developer.android.com/training/permissions/requesting#explain>; accessed September 30, 2021

has shown that subjects tend not to revoke any consent afterward (which is true for 85 % of the users). In the past, this can have led to risks if the functionalities of an application have changed, but the permissions were requested early. Therefore, Android has introduced an automatic deletion of permissions with Android 11 (API level 30) if the application has not been used for a while.

Furthermore, official repositories such as the Google Play Store suggest that applications are secure. However, this is not the case and can be misleading to a user [273] who eventually introduces false assumptions into the information disclosure process (c.f. Section 1.2.3).

Official sources as false friends

User-based Privacy

3.2.2

On the user side, the given construct of the privacy paradox occurs, explained in detail in Section 1.2.3. The desire to use an application may be great and outweigh any concerns, i.e. the privacy calculus is relevant in this context. Recall that people make decisions based on different antecedents. The three aspects of the triangle are also applicable here and distort the picture accordingly. As we learned, an application must request runtime permissions when a specific action should be performed. Thus, it may ask the user to permit, for instance, access to short message services to automatically receive and process an access code to confirm an account. Although a user can manually copy and paste this code into a specific application, it might be *convenient* to give the application that specific permission and allow it to perform the task [110, 188]. This is just one example of beneficial-oriented motivation for information disclosure; personalization may be another here [202].

Convenience

Another key term in this context is *bounded rationality* as defined in this work. Individuals are therefore unable to derive the best decision in terms of information disclosure and the decision of permission choices. The asymmetry of information regarding individuals and developers (including sophisticated tech companies such as Google and Facebook) makes the calculation of benefits versus costs unpredictable since the consequences (i.e. costs/risks) of the disclosure are unknown, highly complex, and can shape later in the future [344]. Applications are subject to being over-permissioned, i.e. they request more permissions than needed [221]. This circumstance is difficult to recognize for a common user. In order to relieve the user of the decision when asking for permissions, Android provides descriptive texts and reasons to be presented to the user as to why a permission is needed. This approach is critical for several reasons. First, an individual must independently understand the descriptive text. Second, the description must be related to the consequences, positive or negative. Third, a

Bounded rationality

user has to decide whether accepting a permission is justified. People tend to struggle to make these complex inferences [127]. Eventually, a trade-off takes place, which is based, however, on incomplete knowledge. People also tend to underestimate their own protection of privacy [2].

Limited
understand-
ability

However, this assumes in the first place that descriptive texts give appropriate indications of the authorizations and classify them. This is not necessarily the case [13]. In addition, people do not tend to read those texts [273]. This is related to framing effects that describe how a permission request is communicated to a user to convince him to approve probably. This can also be encouraged by targeted nudging or other anti-patterning, further distorting the picture. Nudging, dark patterns, or friction are similar to the framing effect as we explained in Section 1.2.3. For example, an application may request multiple permissions at once at the first startup, which is considered a dark pattern and not recommended by Android guidelines. However, it is common for applications to perform like this to e.g. prevent any downgrade in the user experience. Accepting all permissions at once might also be related to the well-known Nothing-To-Hide mentality defined by Solove [350]. Also, this is related to not being able to estimate the potential consequences of a specific information disclosure move (i.e. bounded rationality). Lastly, the context of a person being asked to decide on a permission request is also important. For example, Germans may be far more reluctant to disclose information compared to Americans. Hence, *culture* must be considered. Of course, this picture can be flipped. People might tend to permit apps from reputable developers because they *trust* them [110].

In general, all the antecedents defined and explained in Section 1.2.3 can be transferred to the field of permissions in Android.

3.2.3 Recapitulation

Coarse
permissions
as limitation

The Android permission system is a fundamental building block to ensure privacy and security. Recalling our privacy definition, privacy should enable a user to decide which information is disclosed how at any point freely. However, this is not completely reflected in the current implementation. In this case, we are not talking about technical issues where the target level defines the available permissions and, thus, the provided security level [9] Instead, only a specific predefined set of permissions exists that is relatively coarse, so deciding which information is shared is also coarse. Next, a developer decides which function requires the set of permissions and thus predefines routes a user can take. The

freedom to choose how information is processed is often limited⁴. Finally, users are considered to be able to decide what is best for them. Due to limited awareness or knowledge in combination with the privacy calculus, more tools are needed.

We conclude that a permission system itself must be accompanied by Privacy Enhancing Technologies (PETs) and Transparency Enhancing Technologies (TETs) to further support a user. Android picked up the idea of Transparency Enhancing Technologies (TETs) in its next version 12 (API level 31) by introducing a privacy dashboard and a camera and microphone indicator, as well as basic Privacy Enhancing Technology (PET) functionality for these two sensors. Within this work, we focus on PETs and TETs in Chapter 13 and present our own PET in Chapter 14.

Conclusions

Android's permission system rarely considers zero-permission sensors. Consequently, users cannot change any of these sensors' behavior nor steer their usage, forming multiple attacks around zero-permission sensors with various motivations, including classification, identification, tracking, and more. Part III will dive deeper into this topic.

Zero-permission sensors

Example: The Bread (Joker) Malware Family

3.3

As mentioned, the applications offered on the official store (namely the Google Play Store) suggest a certain level of authenticity and trustworthiness [273]. The following example is intended to show that dangers can nevertheless arise from such applications that are distributed via official resources and also use defined APIs. Consequently, they are also subject to the Android permission system. However, as the Bread (or Joker) malware has shown, threats will nevertheless arise and will require consideration also on the system side, i.e. Android.

The Bread (Joker) malware family was a large-scale billing fraud first detected in 2017. The malware initially carried out Short Message Service (SMS) fraud but quickly adapted to changes in restrictions and switched to Wireless Application Protocol billing fraud. Wireless Application Protocol is a technical standard and protocol suite for accessing the information on a mobile wireless network. It was common in the 2000s with mobile devices having subordinate capabilities in contrast to recent devices fully able to access standard web services. Joker applied many different cloaking and obfuscation techniques to circumvent system

From SMS to WAP fraud

⁴For example, a user might want to request the closest restaurant to his location and therefore allows the application to query his current position via GPS. The application is now able to derive the information for any other purpose

restrictions, including Google Protect, which scans apps in the Play Store for malware.

Course of
the attack

The Bread (Joker) malware family originally performed **SMS billing** by sending a SMS to a predefined number providing a prescribed keyword. The user's carrier then adds the charge for the "subscribed" service to the bill. In addition, the malware was later able to perform **toll billing**. A user visits a website to complete a payment process for some service. Then the payment is processed if the user connects to the website via a mobile network, i.e. the carrier can identify the user and add it to the bill. Alternatively, malware triggers a confirmation SMS on the subscription service and processes it automatically to finish the process, eventually resulting in billing from the network operator. Interestingly, the malware was also able to automatically solve the captchas experienced in the subscription process. Both attacks circumvent the need for entering any payment information, such as credit card numbers. In particular, the workflow legitimizes the device, so a carrier will start collecting subscription fees, although the user never legitimized it.

Hide and
seek game

The Bread (Joker) infection is a two-step process. The diversion application does not initially contain any malicious code other than a downloader. After downloading a malicious application from official sources like the Google Play Store, the infected application initializes itself and starts the fraud by dynamically loading malicious code in the form of a DEX file (compiled Android application code file) from a command and control server. The malware also receives dynamic code and instructions over Hypertext Transfer Protocol and executes them using JavaScript-to-Java callbacks. Apart from this technique, multiple other obfuscation methods for strings, data, and API-calls as further layers of protection against static analysis are in place. Guertin and Kotov [156] gives more detail on how they were used. The malware family attacked only specific countries and thus holds a list of specific mobile country codes with a control server providing respective information to subscribe to premium services. If this verification is successful, the previously described steps are commenced. The victim must have a Subscriber Identity Module card present, most likely due to its function.

Countermea-
sures by
google

As illustrated, Joker successfully bypassed many protective measures in place without the user having any chance to interfere or at least notice any malicious behavior. Consequently, Google adapted its Play Store workflow to provide a safe environment to distribute applications. This is essential to be in line with the subconscious expectation of users [273]. First, the apps were removed from the store. Then the guidelines for two crucial permission groups (a set

of permissions), namely call and SMS, were changed so that developers have to declare their usage before submitting the application to the store [33]. In addition, Google streamlined the workflow of receiving confirmation SMS in some use cases by intercepting them on the system side and forwarding them to an application. This removes the need for an application to request a `READ_SMS` permission.

Modern smartphones are equipped with extensive sensor capabilities that can be processed performantly within the device domain and thus efficiently support new use cases economically. Furthermore, data from smartphone sensors are inherently comparable in fault tolerance to dedicated measurement devices [131], further lowering barriers to entry for new applications. This chapter presents a framework that supports data acquisition specifically for the issues raised in this work. It addresses the particular challenges and problems of Chapter 2 by providing a holistic platform for collecting sensor data with Android devices.

To grasp the challenges and difficulties of speed estimation, some fundamentals of velocity calculation will be discussed in the following. The velocity v of a moving object can be determined based on the GNSS sensor such as GPS (the terms speed and velocity are used synonymously in the context of this work, even though velocity also considers the direction of the movement). It is obtained from the distance between two subsequential GPS locations (loc_{t-1}, loc_t) and their respective measurement time difference $\Delta\mathcal{T}$:

Gps-based
estimation

$$v_t = \frac{loc_t - loc_{t-1}}{\Delta\mathcal{T}}$$

The Haversine formula may be used to calculate the distance between two locations, as Earth is an irregularly shaped ellipsoid. Modern mobile OSs such as Android and iOS provide the measured speed comfortably. However, sustained use of GPS is energy-intensive, which is not suitable for a mobile environment with limited battery capacities [44]. Thus, alternatives to eventually reduce the energy footprint are of interest (see also Section 12.2.2).

Acknowledgement

Parts of the research presented in this chapter are based on supervised work [S8, S9, S11].

Given an acceleration acc_i and a time interval $\Delta\mathcal{T}$, the velocity of an object at time t_i can also be determined with

$$v_{t_i} = v_{t_{i-1}} + acc_i \Delta\mathcal{T}$$

with $v_0 = 0$. While the time difference from the last known velocity $v_{t_{i-1}}$ to the current reading is exactly recorded with each recording, the acceleration has to be handled with care.

An accelerometer outputs the acceleration of an object along all three axes x, y, z , including the omnipresent gravitational force of high-magnitude that falsifies the velocity derivation of an object. Hence, the sensor outputs have to be isolated from disturbances to get meaningful readings, eventually reflecting only the movement of the said object. However, further disturbances such as the engine's vibrations, centrifugal forces, or even the quality of the sensor itself induce other errors that will degrade the velocity estimation. They have to be filtered out and handled with care to enable the proliferation of the sensor readings as error propagation has to be taken into account: Every error will ultimately be present in all successive readings as each velocity relies upon its predecessor. Consequently, processing such as filtering or reshaping new measurements has to be conducted adequately and carefully in order to be able to record as accurately as possible the actual acceleration of the object under investigation.

alignr In this section, we present *alignr*, a holistic approach that considers the uncertain environment of low-quality, error-prone sensor data, complex traffic settings, and device-dependent biases. It addresses the mentioned difficulties based on proven approaches from related work, ultimately integrating them in a retrofittable Android module.

In the following, we present

- ▶ an introductory SLR that identifies related works and presents a taxonomy of building blocks for velocity inference and error handlings built upon the findings,
- ▶ an analysis of sensor data and respective errors when working with IMU produced measurements,
- ▶ an approach applying smartphone-to-vehicle alignment including a Proof-of-Concept (PoC) architecture to derive the velocity of a vehicle robustly, and
- ▶ a comprehensive evaluation of the performance and accuracy of the presented method based on real-world tests.

First, a survey is conducted in Section 4.1 to derive an overview of methods for velocity estimation based on sensor data. The insights are discussed in Section 4.2. Based upon, Section 4.3 presents another proposal for the given task that takes into account learnings from related work. Section 4.4 introduces a retrofittable API in form of a PoC that specifies the technical implementation whose details are illuminated in Section 4.5. A thorough evaluation and performance assessment follows in Section 4.6. This chapter is concluded in Section 4.7.

Structure

Structured Literature Review

4.1

A brief analysis of related work in the form of a literature review is intended to provide an overview of existing methods for inferring the velocity based on IMU sensor data. First, appropriate research questions are defined, followed by an explanation of the methodology used. After that, an overview of the document corpus is presented.

Research Questions

4.1.1

The purpose of this review is to derive building patterns for a PoC application that should i.a. serve as a data collection utility for further research conducted in this work. The question of interest is an understanding and assessment of how velocity can be collected based on IMU data. This yields two research questions:

- Q1** What approaches and techniques exist to derive the velocity based on IMU data? What sensors are of interest?
- Q2** How are errors within the works handled w.r.t. the deflection and correction methods known from Section 2.3.2?

Search Process

4.1.2

The search process for the SLR is based on Kitchenham and Charters [209] which requires fitting the research question, clear inclusion and exclusion criteria, and a definition of the search space, as well as an evaluation with sorting of the identified work.

We selected ACM Digital Library, SpringerLink, IEEE Xplore, Elsevier Scopus, and ScienceDirect as libraries to look for relevant research. This study focuses on approaches that can be executed with the support of a smartphone, consequently excluding works that are not feasible given this constraint. Therefore,

Search string

the following search string¹, based on the variable for the approach and one for the recording device, was applied to all libraries.

```
[ SPEED | VELOCITY ]
& [ ESTIMATION | INFERENCE ]
& SMARTPHONE
```

Criteria A year range limitation was not specified, but the setting with smartphones or mobile devices naturally created a youthful study topic. We allowed ourselves to include work that looked relevant even if it had not been detected using the stated approach (i.e. forward search and backward search) but deleted any duplicates.

To find relevant work, we created the following inclusion and exclusion criteria. To address the research questions, all articles that met the requirements were thoroughly evaluated, including a full-text read and discussion.

1. The proposal tries to infer the velocity based on sensor data not required to be produced by smartphones.
2. A GNSS sensor is not the main source of information for further processing.
3. The solution has to be explained in a comprehensible way to extract the corresponding information.
4. The work proposes a new approach to the problem and is not of type survey or overview.

Only works in English that are peer-reviewed and accessible to the authors were considered. No quality assessment was performed as described by Kitchenham and Charters [209] as only little work was identified. The literature was screened until the date of the SLR (June 2021).

4.1.3 Relevant Findings

Thirteen publications were selected given the search term and criteria from an initial corpus of 9884 findings using a semi-automated, multi-step process. We excluded two publications in favor of their extended versions. The document corpus, including the publisher and the field of research evaluated, is shown in Table 4.1. From the number of results, one may conclude that the research field is not very stimulated; however, there is a steady number of developments. Most

¹For an explanation of the notation, see Appendix B

works were found at IEEE, followed by ACM and IET (via backward search); Springer is seen in one case. The aggregated field shows that mobile computing is leading, including publications focused on sensors. Due to the proximity to mobility, works can also be found in the field of vehicular topics, such as *IEEE Intelligent Vehicles Symposium*.

Overview of the 13 publications identified in the SLR. The works are assigned to different disciplines. **Table 4.1**

Publication	Year	Publisher	Field
Gu et al. [155]	2019	IEEE	Mobile Computing
Bagheri et al. [30]	2018	IET	Mobile Computing
Wang et al. [391]	2017	IEEE	Vehicular
Kang and Banerjee [199]	2017	IEEE	Vehicular
Yu et al. [417]	2016	IEEE	Mobile Computing
Lindfors et al. [234]	2016	IEEE	Vehicular
Ghose et al. [144]	2016	ACM	Mobile Computing
Chowdhury et al. [82]	2016	Springer	Mobile Computing
Leakkaw and Panichpapi- boon [227]	2014	IEEE	Mobile Computing
O’Kane and Ringwood [285]	2013	IET	Mobile Computing
Fazeen et al. [125]	2012	IEEE	Vehicular
Wu [406]	2011	IEEE	Systems Design
Chandrasekaran et al. [70]	2010	ACM	Mobile Computing

Analysis of Speed Inference Methods

4.2

Based on the document corpus, the research questions will be answered and discussed below in this section.

Overview of Methods (RQ1)

4.2.1

First, the portfolio of the methods will be considered, that is, which sensors are used. Then we take a look at the timeline to understand the influence of the

various works to identify appropriately resilient ones. Last, the availability of the approach in the form of retrofittable code or an app is explored.

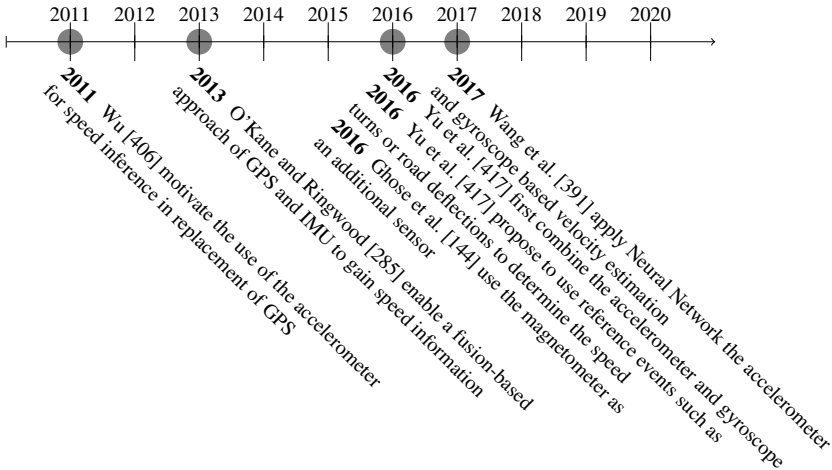
Table 4.2 Relationship of approaches for speed inference and the specific sensors used as a data source. The accelerometer and gyroscope respectively are the most common sensors.

Publication	Accelerometer	Gyroscope	Magnetometer	GNSS	Camera
Bagheri et al. [30]	●	●	●		
Chandrasekaran et al. [70] [†]					
Chowdhury et al. [82]	●			●	
Fazeen et al. [125]	●				
Ghose et al. [144]	●	●	●	●	
Gu et al. [155]	●	●			●
Kang and Banerjee [199]	●	●		●	
Leakkaw and Panichpapiboon [227]	●				
Lindfors et al. [234]	●				
O’Kane and Ringwood [285]	●			●	
Wang et al. [391]	●	●			
Wu [406]	●				
Yu et al. [417]	●	●			
<i>Sum</i>	<i>12</i>	<i>6</i>	<i>2</i>	<i>4</i>	<i>1</i>

[†] Paper relies solely on the signal strength

Applied
sensors

The sensors employed are assigned to the respective articles in Table 4.2. All methods have in common that they work based on the accelerometer and process corresponding measured values. This is obvious because of the physical causality of acceleration and velocity. The work of Chandrasekaran et al. [70] holds a unique position as only the signal strength of a cellular network is used as a basis for detection. Looking at the combinations, it is noticeable that the accelerometer is often combined with the gyroscope, the second most common sensor. All the newer works use this sensor, which could be due to their prevalence and presence in smartphones. One work also uses the camera, and thus cannot be considered a permissionless application, just like GNSS-based works (c.f. Section 3.1).



Timeline with noteworthy publications in the field of velocity estimation based on IMU sensor data. Figure 4.1

The latter class of works [82, 144, 285] focuses on improving the GPS signal and simply fuses the sensor data accordingly². Lindfors et al. [234] require the sensors (accelerometer) to be attached to the chassis of a vehicle, inducing additional expenditure.

Before 2011 and the work of Wu [406], speed estimation approaches (e.g. [70]) were present that used the cellular network as a source of information. Specifically, they used the handover process to determine the area and a related road segment where the handover occurred. Using this information and a second handover location, the distance can be calculated, and eventually, a (rough) speed estimate can be made [70]. With reference to Figure 4.1, 2016 was a dominant year in the context of sensor-based velocity estimation. First, events were introduced [417] and the gyroscope was also considered in newer approaches [417]. With the ongoing development of ML, Neural Network (NN)-based approaches were seen in 2017 being projected on the problem of interest [391].

Timeline

² Similar approaches in the field of trajectory reconstruction are shown in Section 12.2.2.

Availability The next property considered is the availability of the various proposals. Existing implementations allow for the direct applicability of suggestions. Especially for ML-based methods, a learning phase is omitted, which has a direct impact on the goodness of the method. Looking at the corpus of documents, it can be seen that for most of the papers (eleven out of 13) found, only a concept or PoC is presented. Thus, such works lack an available implementation. Two articles mention the availability as an application [30, 199]. In particular, no open-source projects or similar are publicly available for embedding the speed estimation approach in own work.

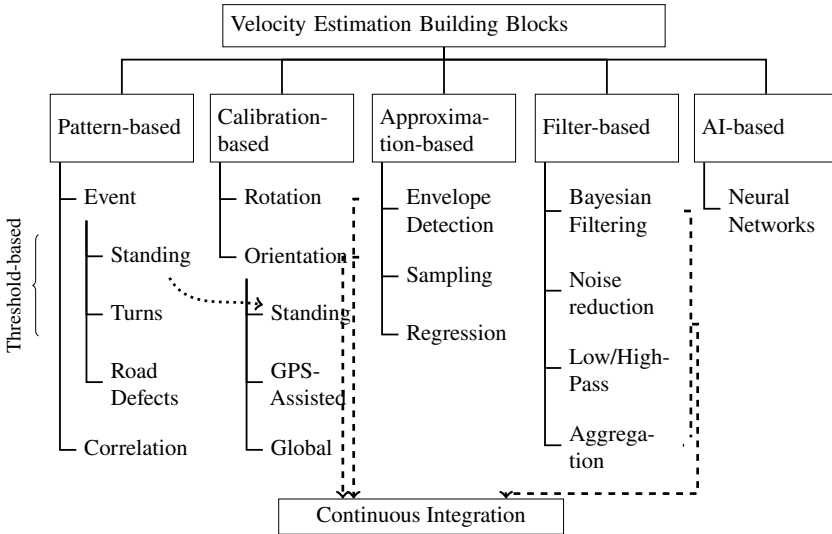
4.2.2 Deflections and Correction Methods (RQ2)

We now discuss the different deflections and error sources discussed within the document corpus' works. Consequently, a taxonomy is built on the document corpus that integrates different approaches and streams for the problem.

Deflections Different types of errors and deflections that eventually impact the velocity estimation were introduced in Section 2.3.2. The types of errors mentioned can also be found in the literature. Table 4.3 considers the errors and presents the implications for speed estimation.

Challenging countermeasures One can use the random walk model to understand these errors and their impact on speed prediction [68]. Second-order random walk-shaped errors such as bias are more challenging than first-order random walk errors, such as noise. This is because a particular type of error propagation also holds for such errors itself: each successive measurement is dependent on the previous state, yielding a dependence on the measurements themselves. Let us discuss an example to understand this issue: A vehicle drives at a specific, roughly estimated speed, then passes through a curve. In this case, a third force comes into play that is measured by the accelerometer, the centrifugal force, which can be considered "bias" in the context of velocity estimation. One has to remove gravity (a constant bias) and centrifugal force (a proportional bias) to get only the acceleration. Gravity is independent of previous measurements, as it is an absolute force. However, the centrifugal force must be calculated based on the vehicle's current speed. Contradictory, this is also only an estimate itself. In summary, each error motivates the need for specific counter-operations to reduce them accordingly to allow accurate velocity estimation ultimately.

Taxonomy of building blocks The next step is to look at how speed estimation is composed in the individual proposals. Based on commonalities, building blocks were extracted, which in combination form a pipeline for speed estimation. The resulting taxonomy is



Taxonomy of building blocks that may be applied to the problem of velocity inference from sensor data. Five different classes are identified with multiple subordinate approaches based on the information found in our document corpus. Event-based and Filter-based approaches are majorly found in the literature. Some approaches are used to reduce errors of sensor data to allow the integration of continuous readings.

Figure 4.2

Identified errors in literature. Error are caused by different defects. Each of them has a different effect on the speed estimation.

Error	Description	Reasons	Effect on Speed	C [†]
Bias	An offset of the sensor readings from different types of sensors that can either be constant, proportional to the measured value, or random.	low sensor quality, missing calibration	An overlaying speed error that grows with each measurement depending on the bias type.	
Constant Value	A sensor outputs the same value over an extended period of time independent of the environment	defective sensor, unresponsive measurement interface	A constant speed for a period of time	●
Drift	A (deterministic, constant) error that grows independently of the measured values over the duration of the measurement process.	defective sensor, low sensor quality	Degradation of the speed estimation accuracy over time	●
Missing Data	No sensor readings are gathered for either a short or extended period of time	defective sensor, unresponsive measurement interface	A constant speed for a period of time	●
Noise	Random measurement values (i.e. white noise) that follow a standard distribution and overlay the actual sensor influences.	low sensor quality, temperature effect	Degradation of the speed estimation accuracy with each measurement value	

continued on next page

Outliers	Harsh, probably short term changes in the sensor readings to values outside of a meaningful range	external impact, sensor defect, temperature effect	Induced offset of the speed estimation
Uncertainty	Unreliable sensor readings are generated	complex environment, external impact	An unknown error of the speed in size and shape compared to the ground truth

[†] denotes if the error is constant over a period of measurements

shown in Figure 4.2. Five basic building blocks that are used for different purposes could be identified.

Pattern-based Approaches First, pattern-based approaches [70, 125, 199, 227, 234, 417] have in common that they try to find patterns in the data from which the velocity is known or from which the velocity can be uniquely determined without depending on previous measurements. In this way, the propagation of errors should be counteracted. *Event*-based patterns can be found in the document corpus. Notable are the standing times, at which the speed can be set to 0 m s^{-1} [199, 227, 417]. All the measured acceleration values acting now come from gravity, which is a perturbation. Yu et al. [417] suggests with curves and road defects (i.e. potholes), two more events that can be used. Thresholds are often defined to detect events; for instance, a standing phase begins when the length of the acceleration vector roughly equals gravity. Event-based patterns are discussed in more depth in Section 4.3. On the contrary, *correlation*-based approaches do not attempt to derive the speed from measurements but use prerecorded and velocity-labeled sequences for comparison [70]. It is assumed that similar measurements yield similar speeds w.r.t. the known data.

Calibration-based The second identified building block is calibration-based methods that target gravity or even the slope of the trajectory, which eventually also alters the gravity impact of different sensor axes. This class can be divided into methods that apply rotation and those that detect the orientation in an initial or ongoing process. To measure the acceleration that reflects the acceleration of vehicles, the axes of the smartphone and the vehicle must be aligned, which is commonly known as a process called “*smartphone-to-vehicle-alignment*”. *Rotation* [82, 144, 199] is performed mainly by moving the three-axis coordinate system e.g. using the gyroscope or rotation matrices (e.g. quaternions). To level the coordinate system, the *orientation* of the device is of interest. The most straightforward way is to use the standing phase, where only gravity acts on the accelerometer [199, 417] and, eventually, the share of the three axes allows conclusions to be drawn about the position of the smartphone in the vehicle. Next, GPS-assisted approaches [82, 144] are also found that use trajectory information to assess the meaningfulness of estimated speeds. Finally, the global approach [30, 144] uses the magnetometer to detect the device’s orientation toward the magnetic north. Subsequently, calibration-based approaches are often used in combination with ongoing rotation methods, as explained, although this is not a general rule [30, 144].

Approximation-based Since the task of estimating velocity is complex, there are other methods that work on approximations. Here, three different approaches are found, which are known from other disciplines like audio signal processing. The basic idea is to build a model that reasonably classifies a course of measured values in order to increase the quality. For example, in *Envelope Detection* [30], maxima are used because they are assumed to approximate the true velocity. Combining these maxima with splines produces the velocity profile that is used as an estimate. Similarly, *regression* models, for example, attempt to form a polynomial [227, 285, 406] that corresponds to the velocity profile. *Sampling* uses intermediate values for the task [144, 227].

Filter-based Filters are commonly seen in literature and are intended to detect outliers (*low- and high pass filter*) or smooth out noisy sensor readings (*aggregation* [30, 417] such as moving average). They are often found as intermediary steps within an approach and are often present in combinations such as Kalman filters combined with low- and high-pass filters. In addition, multiple low- and high-pass filters are combined [30, 125, 144, 285]. *Bayesian filtering* as a category is dominated by the Kalman filter found in many works [234, 285, 406]. Considering [219], filter-based approaches should be used carefully as different devices may require specific parameter settings. It is striking that different works rely on different smartphones within the evaluation such that filtering approaches are not comparable. Furthermore, slight variants in the sensors readings may have an extensive impact on further velocity estimations because the error propagation problem is not be leveled using filters. That process might be challenging. Bagheri et al. [30] uses *noise reduction* techniques known from audio processing for filtering, namely Environmental Noise Cancellation, but this method requires additional knowledge of the type of noise that is being removed from the measurements.

AI-based Last, also Artificial Intelligence (AI) methods, in particular, *Neural Network (NN)* are found. The works [155, 391] both rely on Long short-term memory (LSTM) networks, as this method can memorize a previous state and is, therefore, feasible to process time series-based data. For this purpose, previously measured values of a few seconds can be provided to the system [155]. The use of LSTM allows the processing of raw sensor data, which are fed into a first layer (input layer). In the training phase, the system learns the correlations between sensor values and speed, although the dependence on the vehicle and mobile device used to record the training and test data remains questionable. Furthermore, a “*mechanism of correcting the accumulated error is expected to be built into the network*” [155] which is a strong statement considering the complexity of the task. In particular, both approaches have the same architecture

with a double layer LSTM, and the same feature inputs (accelerometer and gyroscope) as [391] is the predecessor of [155] (hence, to be precise, NNs are only found a single time for velocity inference).

Conclusions
from the
taxonomy

The identified building blocks aim to increase the quality and accuracy of the measured sensor values, each specifically fitting for a particular set of errors. This is particularly true for elements of the calibration and filter class. The relationship between errors and potential building blocks is shown in Table 4.4. Then, these preprocessed sensor values are then i.a. integrated with the classical approach using the known formula for velocity.

Table 4.4 Relationship between errors and the specific building blocks. The building blocks are used to counter the identified errors, with overlaps in terms of methods.

	Building Block						
	AI-based	Filter-based	Approximation-based	Calibration-based > Orientation	Calibration-based > Rotation	Pattern-based > Correlation	Pattern-based > Event
Bias	●	●	●	●	●	●	●
Constant Value			●			●	
Drift		●					●
Missing Data			●			●	
Noise		●		●			●
Outliers	●	●					●
Uncertainty	●	●					

Real-time
suitability

The methods can be further divided into real-time capable methods [30, 82, 125, 144, 155, 199, 285, 417] that can directly estimate a velocity during a recording and those that calculate the speeds *ex-post* after completion of a trip [30, 70, 227, 234, 391, 406, 417]. The latter methods usually show more accurate results since, for example, detected events in the sense of backtracking can also affect previous measurements. This is also in line with statements about hybrid velocity estimation approaches, which are capable in real-time as well as *ex-post* capable. These perform better in the latter operating mode [30, 417].

A Retrofittable Take on Speed Inference

4.3

First, such a proposal should consider the arbitrary positions of a smartphone within a vehicle, which is not the case in some works. Thus, smartphone-to-vehicle alignment must be present and work robustly, as it is crucial for further processing. To meet this requirement, methods from the calibration building block are suitable. In particular, we will measure the orientation in the stationary position via an accelerometer because there, as already described, only gravitation is effective. Using standing phases, of course, implies that it can be detected with high confidence. The gravitation is successively rotated with the help of the gyroscope (rotation-based approach) so that the respective fractions are distributed on the axes according to the *momentary* orientation to allow for a slope-aware alignment [199]. Therefore, we also rely on approximation-based methods as the rotation can only be sampled in a non-continuous manner.

Orientation

Next, noisy and faulty sensor data should be handled accordingly to overcome the estimation degradations illustrated in Table 4.3. Especially outliers need to be detected, and their impact on the velocity inference has to be reduced. The quality of the sensors must also be taken into account accordingly so that drift, for example, must be detected and compensated. In the best case, an algorithm can independently detect constant error variables and evaluate them accordingly. Therefore, filter methods from the respective building block will be implemented; hence, they must be applied carefully, not falsifying the measurements.

Uncertain
sensor
quality

The uncertainty of the whole process should be considered by detecting and evaluating events found in measurements similar to Yu et al. [417] that will function as reference points where the fragility of the entire estimation is reduced. The process of event detection can yield meaningful results, as shown in the literature, but may be resource-intensive, especially when performing sophisticated pattern recognition. As a smartphone only poses a constrained environment with limited computing performance, an approach must be specifically optimized for such a context to enable the real-time processing of sensor data.

Reference
points

To enable real-time usage of the approach, it must be retrofittable in known ways to ultimately represent a drop-in replacement for speed acquisition. There, appropriate APIs need to be available that are similar to the ones known from working with GPS-based data. Furthermore, the proposal must be accessible e.g. as open-source.

Retrofittabil-
ity

In the following, we will present an analysis of the applied building blocks and how they will be included in the proposal. Figure 4.3 posts that pattern-based

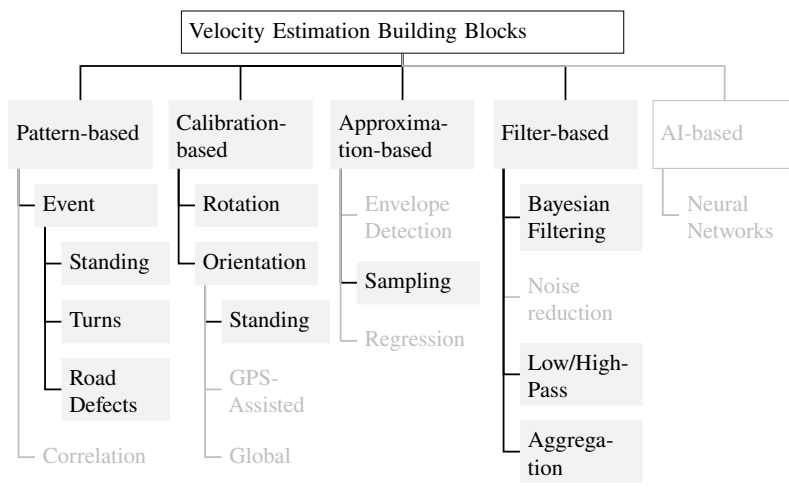


Figure 4.3 Applied building blocks within the proposed approach. The taxonomy (c.f. Figure 4.2) from the literature review is used to create a superset of methods to integrate into an holistic approach.

elements will be found, as well as calibration-based ones, and eventually filter-based methods.

4.3.1 Overview

Our proposal for velocity inference based on sensor data is called *alignr*. It incorporates multiple building blocks of the taxonomy to address the initial problems. The architecture and flow of the process are illustrated in Figure 4.4.

Flow *alignr* uses data from the accelerometer and gyroscope and acquires it using a standardized APIs, i.e. the Android `SensorManager`. This data is then cleaned from gravity, which is a significant disturbance. The data cleaning process is accompanied by a continuous rotation of the coordinate system of the mobile device that records the data. The objective is for both the smartphone and the car to have the same orientation in a three-dimensional space so that the sensor data recorded by the smartphone represents an approximation of the forces acting on the vehicle. Successively, events known from the literature are extracted from the data, and the speed is estimated either based on identified events or in a continuous manner using the integration method introduced at the beginning of this chapter. Multiple modules then process the derived speed to minimize the

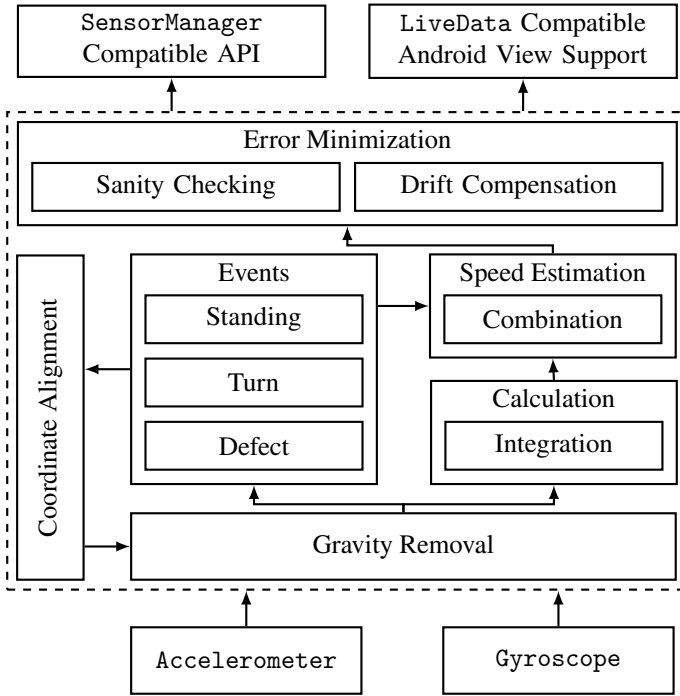


Figure 4.4

Overview of the design of *alignr*. Data is gathered using in-built sensors and then preprocessed to reduce noise and bias. Successively, events are extracted from the data, and the speed is estimated either based on identified events or in a continuous integration manner. In parallel, the die coordinate system is aligned with one of a vehicle. Error minimization is performed to increase the velocity estimation accuracy. The approach can be accessed using a `SensorManager` compatible API or directly integrated in an Android application using `LiveData` support.

error. Here, sanity checks are performed to address the problems of uncertainty, drift, and bias.

A fundamental requirement for *alignr* is the ease of use and the ability to embed it in apps. For this purpose, the approach can be accessed via a `SensorManager`-compatible API that extends the well-known Android interfaces to also capture velocity similarly to other sensor resources. Additionally, the velocity estimation

Employing *alignr*

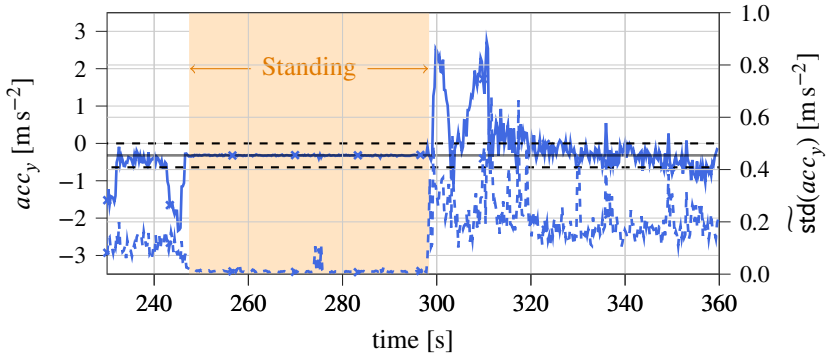


Figure 4.5 Example of a trajectory with a standing phase. During the standing phase, the fluctuations measured by the accelerometer decrease on all three axis and stay within a certain threshold for an extended period.

and associated parameters can be integrated directly into an Android application that provides support for LiveData³.

4.3.2 Events

By including events in the velocity calculation, the quality of this estimation can be increased. Events allow the calculation and conclusion of the velocity without the problem of error propagation, which is known to introduce a large uncertainty into the calculation. Likewise, the influence of disturbance variables such as gravitation is reduced, and alternative computation methods than the integration approach are resorted to. In the following, three known events from the literature are considered, namely, detection of the standing phase, impact of turn events, and evaluation of road defects [417].

Standing Phase

Due to various factors, such as bumps on the road, the movement of a vehicle causes an increase in noise, which is detected by the accelerometer. However, once a vehicle comes to a stop, the sensor noise drops dramatically and moves in a narrow band around the rest position. This condition is shown in Figure 4.5 that

³ LiveData enables to continuously monitor (fast-)changing data structures and eventually updates a user interface efficiently and effectively. It is aware of the current state of an application and the recommended way to bind data types to components in the user interface; see <https://developer.android.com/topic/libraries/architecture/livedata>.

illustrates the measurements of the y-axis of the accelerometer (acc_y). The color-coded area symbolized the phase when the velocity decreased to 0. It is initiated by negative readings, representing a braking maneuver, and delimited by a solid positive deflection, the subsequent acceleration. This phase is of fundamental importance in the approach because it is the only phase that allows for a reliable and confident estimate of the speed.

The reduction in noise that defines a standing phase can be seen on all axes. Therefore, the accelerometer is continuously analyzed on all three axes by calculating a short (e.g. 1 s) rolling standard deviation std based on the length of the accelerometer vector $|acc|$. If this value falls below a certain threshold $\theta_{S,\sigma}$, then a standing phase is assumed. If $\theta_{S,\sigma}$ is exceeded again, the velocity integration continues. Even though gravity is included in this process, it is static and, therefore, does not matter.

Detection

It is crucial to balance between false positives and false negatives. A too-late exit from the standing phase (false positives) leads to the fact that initially strong acceleration, as shown in the figure, is not fully considered in the velocity estimation and inevitably leads to a deviation that affects all future values. The same applies to the false detection of a stationary phase in the middle of a regular trip. Thus, the system is designed to avoid false positives but at the same time accept false negatives. These have little effect on the speed estimation and lead only to negligible deviations from the estimated to the actual speed, which is zero. This is due to a short phase of velocity integration during false-negative detection, during which, additionally, usually no significant forces act on the accelerometer.

False positives

Turn Events

A turn can be detected using the gyroscope, as this sensors measures rotation around all three axes. Hence, the z-axis (c.f. Figure 2.3) will yield information about the rotation of the vehicle e.g. while turning. The relationship between a turn and the respective measurements on gyr_z is shown in Figure 4.6, where in total four turns are present. Sharper turns yield higher peaks with more steering involved, and the direction of the peak represents the direction of the turn.

This sensor is independent of the current speed of a vehicle and any acceleration that impacts it. However, another force is introduced when turning, namely the centripetal force, which is picked up by the accelerometer. This force can be calculated using the gyroscope and the accelerometer to gain the tangential speed of a vehicle, as shown by the relationship in Figure 4.7 [393, 417]. Let r be the

Derivation of velocity

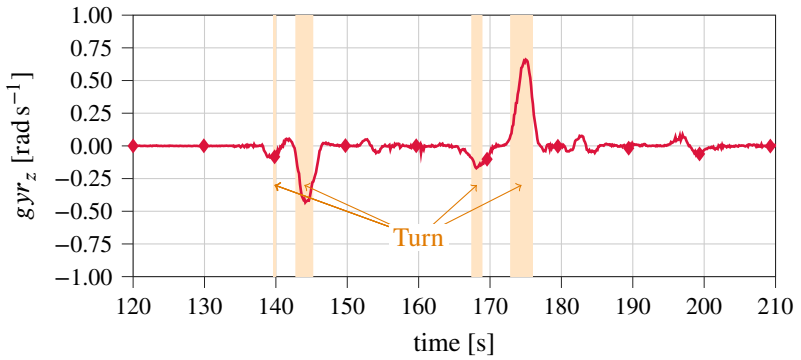


Figure 4.6 Example of a trajectory with four turns. Turns can last for different lengths of time and reflect different changes in direction.

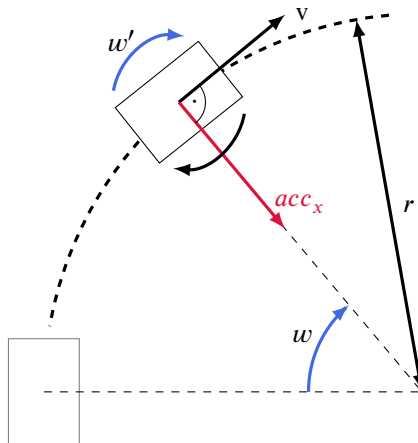


Figure 4.7 Relationship between the centripetal force, the tangential speed and the speed of a vehicle. (based on Wang et al. [393] and Yu et al. [417])

radius of the turn, w the angular velocity, and acc_x the centripetal acceleration. Then the vehicle velocity is defined per

$$v = wr = \frac{acc_x}{w}$$

This expression assumes that the angular velocity w induced by the turn is equal to the angular velocity of the vehicle, denoted as w' . Furthermore, the unknown radius r of the turn can be substituted by $r = acc_x/w^2$.

The speed derived by turns does not rely on previous speed estimations. Hence, it can be used to eliminate potential deviations induced by the integration-based approach. This is a sampling-based method, as the current velocity of the turn can only be calculated once new measurements of the accelerometer and gyroscope arrive. However, we empirically found that early turn measurement values show speeds that are typically too high. The estimated velocities become more accurate with the forthcoming progression of a turn, so later measurements are weighted higher in the calculation. In addition, the accuracy of the previous method depends on the turn speed, with slower speeds resulting in more accurate predictions [417]. To derive the velocity based on this event, the sensor stream is continuously observed for any occurring turns. Later, we also introduce some optimization to only select feasible turns for calculation and thus reduce the impact on the system.

Noteworthy
details

Road Defects

Additionally, the proposal to use road defects from Yu et al. [417] is adapted but extended to be feasible for real-time processing in the capability-limited smartphone environment. In general, time series data may seem random, but reoccurring patterns are present as time series, in our case, are generated by sensor data that, apart from bias and noise, are influenced by the driving behavior and eventually road conditions. When a vehicle passes a road defect, such as a bridge joint, a repetitive pattern is created that becomes observable via the z-axis of the accelerometer. Such a sample is depicted in Figure 4.8. It is formed when the vehicle is exposed first to an impact on the front axle, followed by the rear axle. Consequently, there are two peaks present that semantically belong together. The time difference Δt of both peaks can be extracted from the measurement data and allows the calculation of the velocity using $v = d/\Delta t$, if the covered distance is known. d , however, can be assumed to be static, as in this case it equals the wheelbase of a vehicle. Noticeably, speed inference is only possible up to specific values due to sensor and physical limits [417]. We also refer to Section 5.4 and Chapter 8 for a more detailed analysis of the derivable patterns.

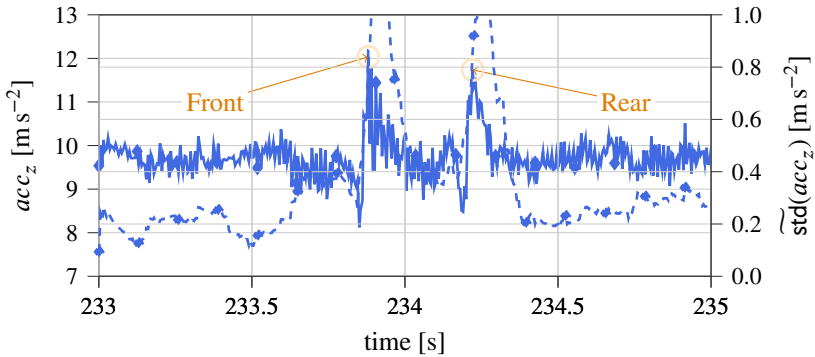


Figure 4.8 Example of a trajectory with a road defects. Two successive peaks are detected that are induced by the front followed by the rear axle passing the same road defect. A peak's highest point used for velocity estimation is only a single measurement.

Matching peaks

Yu et al. [417] uses autocorrelation to match two related peaks, which requires continuously calculating the similarity between two segments. Normalized autocorrelation yields a number in the interval $[0, 1]$, with nearly identical segments closer to 1 and dissimilar segments closer to 0. That is, the current segment where a peak was detected and previous (or lagged) segments in the time series measurements up to a certain threshold $\tau_{B,l}$ posting a similar peak. This requires calculating the correlation of the time series with a copy of itself as lags (i.e. delays) of $[\pm 1, \pm 2, \dots, \pm \tau_{B,l}]$. The time distance between the two peaks is the lag with sufficiently high similarity. Therefore, this delay is $\Delta t \leq \tau_{B,l}$.

Simplification of the detection

Notably, the calculation of the correlation up to $\pm \tau_{B,l}$ must be faster in sum than the acquisition frequency of sensors measurements for it to be applicable. A property of road defects is exploited to overcome the computationally intensive task of comparing similarities within a high-frequency time series data stream. Autocorrelation can occur only if a peak is detected; only then does the need to search for a second one arise to yield the pattern of interest. Therefore, a short std of acc_z is kept; if there is a rash that justifies a deeper analysis, autocorrelation can be triggered.

4.3.3 Coordinate Alignment

To derive meaningful events as described previously, an aligned mobile device is assumed with its axes being congruent with those of the vehicle. This task

is known as smartphone-to-vehicle alignment and was already mentioned in Chapter 2. Now we explain how to align the smartphone and vehicle based on Wang et al. [393] who employ a rotation matrix to achieve alignment, referring to calibration-based building blocks. It should be noted that this yields an initial orientation of the vehicle and smartphone that is based on the accelerometer and gyroscope. It does not take into account potential changes between the relative orientations of both, although the position of the device may change slightly over time, for example, caused by driving maneuvers. However, it is sufficient to assume that the mobile device is not moved in addition while driving, as it is forbidden in most countries to use the smartphone while driving. A static position in the car, e.g. in the can holder or on the dashboard, is assumed, yet it can be of arbitrary orientation. We refer to this statement throughout this work multiple times. In the end, the coordinate alignment task has to be repeated from time to time.

First, a rotation matrix \mathcal{R} is constructed [393]

Rotation
matrix

$$\mathcal{R} = \begin{bmatrix} x_i & x_j & j_k \\ y_i & y_j & y_k \\ z_i & z_j & z_k \end{bmatrix}$$

where $\langle x, y, z \rangle$ represents the coordinate system of the vehicle with the unit vectors $\langle \vec{i}, \vec{j}, \vec{k} \rangle$ (c.f. Figure 2.3). If the devices are perfectly aligned, the matrix formed by the three vectors $\vec{i}, \vec{j}, \vec{k}$ will be an identity matrix (i.e. $\vec{j} = (0, 1, 0)^T$). However, in reality, this is rarely the case, as the $\langle x, y, z \rangle$ axes of the smartphone equalling the ones of the vehicle. Therefore, each vector has to be determined to eventually use \mathcal{R} to rotate sensors readings by the smartphone back into the vehicle coordinate system.

A process that has proven its feasibility in finding the unit vectors is as follows.

Finding
unknown
parameters

\vec{k} The accelerometer measures gravity *gra* once a vehicle is standing still, which is assumed to be the z-component of a vehicle. Therefore, \vec{k} can be determined by evaluating the sensor readings during this period, probably using averaged values. The vector as the output of the current readings of the smartphone's accelerometer is normalized and smoothed using a low-pass filter to set it as \vec{k} . A fundamental assumption is that the whole gravity also impacts solely the vehicle's z-axis, which is probably not the case due to sloppy roads or uneven surfaces. To overcome this drawback, we introduce a reoccurring determination

of \vec{k} and dependent variables at each stop. We empirically noticed that the vehicle's orientation during a trip is even.

\vec{j} Within the coordinate system, j equals the y -axis, thus the vehicle's acceleration. The linear acceleration virtual sensor (that is based on the accelerometer but removes gravity and other noise⁴) is used to detect the orientation by observing the proportion of the acceleration solely from the vehicle impacting the smartphone. This only yields meaningful results if and only if the force acting on the smartphone is due to the vehicle accelerating or braking. Hence, we refer to the gyroscope to only collect acceleration readings when true linear movement is present, i.e. the gyroscope does not measure any significant rotation. This rules out any centrifugal force biasing the readings. On-going readings are collected and aggregated as soon as no rotation is picked up to improve the accuracy of \vec{j} .

\vec{i} In the system of equations based on three variables, the last unknown component \vec{i} can be calculated by the cross product of \vec{k} and \vec{j} .

Continuous
rotation of
sensor
readings

The rotation system \mathcal{R} is accurate after determination (approximation-based approach). With further progression of a trajectory, it may become inaccurate due to environmental impacts on the vehicle or smartphone. Therefore it is important to readjust it whenever possible to effectively rotate sensor readings from the smartphone into the vehicle coordinate system.

4.3.4 Rotation-based Gravity Removal

Even though gravity should only be present on the z -axis due to the rotation of the coordinate system within the framework of smartphone-to-vehicle alignment, this assumption is unrealistic in reality. Therefore, it must be continuously removed from any measured values. The goal is that the y -axis measures acceleration or braking events while acc_x represents the centrifugal force, which is necessary for the curve-based speed estimation. Gravity should be applied exclusively to the z -axis.

Android-Provided Functionality

Android provides virtual sensors that address the problem. However, as it becomes evident, these sensors are too coarse to be used for speed estimation.

⁴We will later discuss this sensor and its inability to be used for velocity estimation because of inaccurate measurements.

Android provides virtual sensors to split the acceleration (TYPE_ACCELEROMETER) into linear acceleration (TYPE_LINEAR_ACCELERATION) and gravity (TYPE_GRAVITY) [102]. A low-pass filter can be and is used to filter out harsh yet short spikes in the acceleration readings that can be attributed to movement, i.e. TYPE_LINEAR_ACCELERATION. The parts that are admitted by the filter are then assigned to be the gravity portion of the current reading. However, the assumption for that calculation will not hold in our case as the acceleration in mobile environments may last for more than a few seconds which ultimately convinces the low-pass filter to assign such parts of linear acceleration to the gravity vector, eventually skewing it⁵. Using such a derived gravity to subtract from the accelerometer will yield too small linear accelerations, making this approach (even with an optimized alpha) not applicable.

Filter-based approach

A second approach that is provided and adaptable for the given task relies on a rotation vector (TYPE_ROTATION_VECTOR) that is a composed sensor [102]. The sensor fuses data from the accelerometer, gyroscope, and magnetometer to provide a measurement of a rotation based on a reference coordinate system where the z-axis always points to the sky and the y-axis to magnetic north (the x-axis can be calculated from both). The output is a quaternion [220] representing the rotation of the device along the axes referred to the reference coordinate system. A quaternion q is defined as $q = \langle \cos(\frac{\alpha}{2}), x \cdot \sin(\frac{\alpha}{2}), y \cdot \sin(\frac{\alpha}{2}), z \cdot \sin(\frac{\alpha}{2}) \rangle$, where α is the rotation around a normed vector $\langle x, y, z \rangle$. For instance, a rotation around the x-axis with angle α yields $q = \langle \cos(\frac{\alpha}{2}), x \cdot \sin(\frac{\alpha}{2}), 0, 0 \rangle$. In conjunction with a known gravity sample, as captured above, this sample can be continuously rotated based on the rotation vector. For this purpose, it is necessary to relate the new rotation to the rotation (i.e. the quaternion) at the gravity sample's recording time. The practical implementation suffers from the problems that can be attributed to the individual sensors from which the rotation sensor infers the quaternion and which are known from the literature (c.f. Table 4.3). These are gyroscope drift, the inaccurate information of the magnetometer⁶, and again the determination of the z-axis based on gravity. Like the filter-based approach, the latter suffers from the limitations of the low-pass filter and the associated inaccurate inference of the gravity.

Rotation-based approach

⁵ Using a low-pass filter is fine for certain use-cases such as the detection if a smartphone is picked.

⁶ We refer to Section 12.6.2; especially in the interior of a vehicle, the magnetometer can only give a rough indication of the direction of the sky, exact degrees are not reliable

Gyroscope-Induced Approach

Another method is to rotate the initially collected gravity sample using the gyroscope continuously. Here, the procedure differs from the rotation-based approach in that it is a relative view of the problem. The gyroscope outputs a rotation along an axis, which can only be interpreted adequately in the temporal course and relies on previous calculations. In contrast, the quaternion of the rotation sensor is an absolute consideration, namely the shift to the reference coordinate system. Therefore, the approach based on the gyroscope is only an approximation.

Quaternion-
based
continuous
rotation

The sample gra recorded during a standing phase is rotated while moving with the help of the gyroscope, and a new gravity vector $\widehat{gra} = gra$ is formed, which can then be subtracted from the total acceleration to obtain only the linear acceleration of the smartphone and thus the vehicle. The gravity at time t is given as pure quaternion $q_{t_i} = \langle 0, gyr_{x,t_i}, gyr_{y,t_i}, gyr_{z,t_i} \rangle$. It corresponds to a point at $\langle gyr_x, gyr_y, gyr_z \rangle$ as $\cos(\frac{\alpha}{2}) = 0 \Rightarrow \sin(\frac{\alpha}{2}) = 1$. This is to be rotated by a unit quaternion that is generated from the current gyroscope measurement $q_{u,t_{i+1}} = \langle 1, \Delta t \cdot gyr_{x,t_{i+1}}, \Delta t \cdot gyr_{y,t_{i+1}}, \Delta t \cdot gyr_{z,t_{i+1}} \rangle$ where Δt is the time since the last gravity rotation and the current measurement. The new gravity is then defined per $q_{t_{i+1}} = q_{u,t_{i+1}} \cdot q_{t_i} \cdot q_{u,t_{i+1}}^{-1}$ which is another pure quaternion, i.e. another vector of the same length. Hence, the gravity as a force does not change, yet it is rotated according to the gyroscope. The initial pure quaternion is $q_0 = \langle 0, acc_{0,x}, acc_{0,y}, acc_{0,z} \rangle$ with the gravity equalling the accelerometer readings gathered in the standing phase (see above).

Euler angles

Quaternions are used in favor of Euler angles to circumvent the gimbal lock problem. The rotational path between two consecutive measurements cannot be determined precisely as the gyroscope only outputs a sampled rotation with limited frequency. However, with sufficiently small time increments (i.e. high frequency) between successive measurements, the assumption of linear rotation between them produces correct *almost* rotations. Due to the energy efficiency of the gyroscope and the accelerometer, high sampling rates can be achieved.

4.4 Proof-of-Concept Application

We now introduce our PoC implementation for Android called *alignr*. It integrates the previous findings and building blocks into a holistic, retrofittable module that can be added to applications that target velocity access using only IMU-based sensor data.

Architecture

4.4.1

The architecture follows a stream-focused data processing approach based on a central event bus whose events can be consumed by different modules. *alignr* is provided as a compatible Android module with no additional permission requirements. It is written in Kotlin and addresses the limited resources found in the mobile environment. In particular, it reduces the number of computations by applying sophisticated and predictive fine-grained filtering methods, eventually using methods known from Complex Event Processing (CEP) (see Sidebar C). The application uses dependency injection through Hilt⁷ to reduce boilerplate code and to manage the lifecycle of an application using Android's built-in methods. Also, the proposal respects the separation of concerns by not requiring any changes within an application's business logic.

Multiple layers organize the *alignr* architecture with multiple patterns driving the design process of the framework. Sensors (in the data layer) generate initial events that are picked up and processed by the corresponding modules. The `EVENTBUS` dispatches information on demand to listeners that perform computation and eventually update the `STATEMACHINE` or produce events on their own. Both are found within the middleware layer. Also, this reduces complexity as neither a global process nor state view is required, and modules can be altered without interfering with others, ultimately providing resiliency. Events (c.f. Section 4.3.2) are extracted in the event layer, while supporting elements, including gravity removal (c.f. Section 4.3.4) are identified in the finders layer. This layer is also responsible for enabling smartphone-to-vehicle alignment (c.f. Section 4.3.3) and eventually speed estimation in the integration layer. All data is stored in a central repository within the persistence layer. Furthermore, an interface layer provides access to the computed speed estimation or any of the intermediate data, such as the smartphone-to-vehicle aligned accelerometer data. A technical overview with dependencies and information flow is presented in Figure 4.9.

Overview

The provided architecture relies on a `STATEMACHINE` as a Single Source of Truth [290]. This is an approach adopted in software development to structure data management, in which a single central component holds data autonomy. All information such as sensor values or current events, as well as speed estimation, refer exclusively to information held centrally in the `STATEMACHINE`. Only at this location can authorized participants change data; local changes within the respective modules are undesired and do not replicate in the system. This

Single source of truth

⁷<https://developer.android.com/training/dependency-injection/hilt-android>

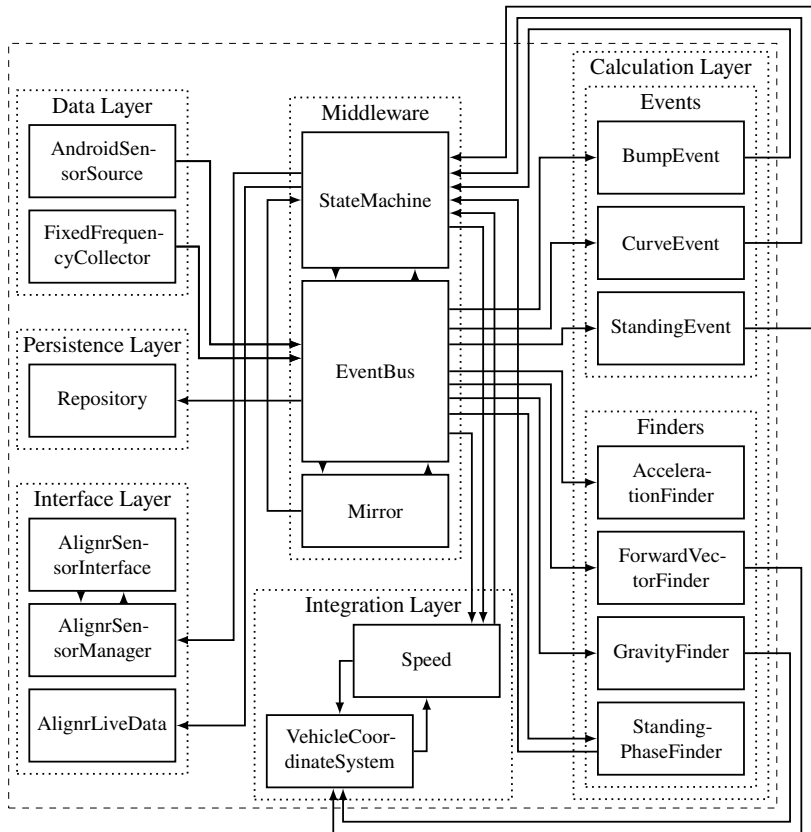


Figure 4.9 Technical overview of the implementation of the *alignr*. Multiple layers and components are responsible for data collection, aggregation, fusion, preprocessing, and filtering as well as estimation and providing access to data. The architecture follows an event bus pattern with multiple listeners that selectively consume events produced by other components. Each component has a dedicated task with reduced interdependencies to reduce complexity and increase modularity.

guarantees an integer processing, with each module (e.g. finder and event) based on the same data work, and the necessity of replication of the data is void. Changes in the STATEMACHINE can be redistributed to all listeners via the EVENTBUS.

The observer pattern is found in the framework enabling one-to-many dependency between objects. However, all components are loosely coupled due to the architecture used, and observers, i.e. components, who are interested in specific status updates, are automatically notified of changes based on their own configuration. In particular, there is no direct update of the state of the components via other components; i.e. all the components are self-isolated. We employ Kotlin's `MutableSharedFlows` and `MutableStateFlows` for this pattern.

Observer
pattern

The proposed architecture is designed to be used within the Model View View-Model pattern. The pattern is used to separate the presentation (`View`) and the processing of the data (`Model`). Data is, per definition, the speed estimation and derived values such as distance traveled. *alignr* also provides support for a `ViewModel` that is used to link a view and a model by offering underlying data through `LiveData`⁸. This is an essential argument for using *alignr* within any application, as the alignment of the smartphone to the vehicle is provided in an easy and comprehensible fashion comparable to the built-in functionality (e.g. `SensorManager`).

MVVM
pattern

Additionally, the repository pattern is used to make the same data sources accessible in different places in the architecture. A repository is used to separate the calculation and estimation of the speed from any data sources. This reduces duplicate code. *alignr* can be used with multiple data sources such as mobile phone sensors or pre-recorded trajectories, but the same computation modules can still be used. Therefore, a middleware adds convenience and utility methods to the different data sources to provide the same functionality across different configurations. The repository pattern is not to be confused with the `Repository` shown in Figure 4.9 that is found within the persistence layer to store data models persistently.

Repository
pattern

API

4.4.2

alignr is intended to provide an easy and accessible API to allow software developers to easily use the Android module combined with sophisticated speed and distance estimation.

⁸ <https://developer.android.com/topic/libraries/architecture/livedata>

```
1 val sensorManager = getSystemService(Context.SENSOR_SERVICE) as
  ↳ SensorManager
2 val accelerometerSensor: Sensor? =
  ↳ sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)
3 val sensorListener = object: SensorEventListener {
4 override fun onAccuracyChanged(sensor: Sensor, accuracy: Int) { }
5 override fun onSensorChanged(event: SensorEvent) {
6     val dT: Long = event.timestamp
7     val axisX: Float = event.values.get(0)
8     val axisY: Float = event.values.get(1)
9     val axisZ: Float = event.values.get(2)
10    // further process values
11 }
12 sensorManager.registerListener(sensorListener, accelerometerSensor,
  ↳ SensorManager.SENSOR_DELAY_FASTEST)
```

Listing 4.1 Illustration of accessing sensor data. A `SensorManager` gives access to a `Sensor` that provides `SensorEvents` with specific sensor readings.

Receiving
sensor
updates

Therefore, it is essential to understand how sensor data is accessed in the Android environment. Listing 4.1 shows an exemplary code snippet to gather accelerometer readings. Android keeps a `SensorManager` that provides access in a generic way to multiple sensors. By implementing a `SensorEventListener`, the appropriate business logic can be deposited to process sensor values. The generic interface has a `onSensorChanged` method that is called with a `SensorEvent` once a new measurement from the respective sensor arrives. A sensor event is a container that provides access to either processed or raw values from the sensor without the need for a developer to apply any conversation. Multiple `SensorEventListeners` can be registered as listeners or unregistered at any point to either receive sensor updates with a user-controlled frequency or stop consuming events.

Challenges

To be precise, Android provides frequency classes dependent on the hardware and sensor employed. For example, `SENSOR_DELAY_FASTEST` requests updates every time the (hardware) sensors yield a value. This highlights a major drawback: Different sensors have different output rates; hence, sensor fusion becomes challenging. Furthermore, the frequency cannot be controlled in a fine-granular way. To handle both problems, we propose a `FIXEDFREQUENCYCOLLECTOR` (see Section 4.5.1) that can collect sensor data from multiple sensors with a well-defined and accurate frequency.

AlignrSensorManager

As the name implies, a so-called `ALIGNRSENSORMANAGER` provides access to sensor data comparable to the Android legacy instance. However, functionality is extended to make all added and derived data available to a developer, including a dedicated speed estimation event. Due to the restricted `SensorEvent`, a `AlignrSensorEvent` will be used as a replacement. This is needed as *alignr* introduces new events with different shapes.

AlignrLiveData

Furthermore, *alignr* provides direct integration into the Android lifecycle by providing `LiveData`. Hence, this data can be directly injected into an Android `ViewModel`. As Android manages `LiveData`, optimization such as battery management is handled by the OS itself. An example implementation is shown in Listing 4.2. `LiveData` is available for almost all computations and events that are seen by the `EVENTBUS` or `STATEMACHINE`. As one can see, basic sensors are also provided as `LiveData` such as the `Sensor.TYPE_ACCELEROMETER`.

The `AlignrViewModel` can be used by an Android `Fragment` to data-bind it to a `View` that provides containers for the `LiveData` values (such as `AlignrFragmentBinding`). Listing 4.3 shows an example.

Flow Overview

4.4.3

The architecture reduces the complexity of data exchange by sticking to an observer pattern. However, different components shown in the architecture pose different requirements, updating states or crafting new events.

Implementation Flow

4.5

Previously, a holistic overview of the *alignr* architecture has been given. This section thoroughly explains the implementation of each layer, including their components that perform dedicated operations, consume and generate events, or update states. First, the objectives are defined, preconditions are addressed, and then the implementation is discussed.

```
1 @ExperimentalTime
2 @ExperimentalCoroutinesApi
3 @HiltViewModel
4 class AlignrViewModel @Inject constructor(
5     val alignr: Alignr
6 ) : ViewModel(), DefaultLifecycleObserver {
7
8     val accelerationVector = alignr.accelerationVector.asLiveData()
9     val gyroscopeVector = alignr.gyroscopeVector.asLiveData()
10    val accelerationNoGravity = alignr.accelerationNoGravity.asLiveData()
11
12    val movementState = alignr.movementState.asLiveData()
13    val curveState = alignr.curveState.asLiveData()
14    val speedState = alignr.speedState.asLiveData()
15
16    suspend fun start() {
17        alignr.start()
18    }
19 }
```

Listing 4.2 Exemplary `AlignrViewModel`. The excerpt shows how to access the events and computations of *alignr* including the speed estimation.

4.5.1 Data Layer

There are two components in the Data Layer that provide the necessary sensor data for the framework. They are the interfaces to the Android API as has been shown in the previous section.

AndroidSensorSource

The `ANDROIDSSENSORSOURCE` component uses standard methods to access and continuously collect data from smartphone sensors to make sensor readings available for further processing within the *alignr* framework. Data from the calibrated accelerometer (`Sensor.TYPE_ACCELEROMETER`) and gyroscope (`Sensor.TYPE_GYROSCOPE`) are collected. Furthermore, the composite sensors `Sensor.TYPE_GRAVITY` and `Sensor.TYPE_LINEAR_ACCELERATION` are used. These are virtual sensors provided by Android that fuse certain physical sensors and perform different preprocessing of the data. The data is gathered at the highest possible frequency (`SensorManager.SENSOR_DELAY_FASTEST`), which varies depending on the Android device and implementation. Based on the

```

1  @ExperimentalTime
2  @AndroidEntryPoint
3  class AlignrFragment : Fragment(R.layout.alignr_fragment) {
4
5      private var binding: AlignrFragmentBinding by autoCleared()
6      private val alignrViewModel: AlignrViewModel by viewModels()
7
8      override fun onCreateView(
9          inflater: LayoutInflater, container: ViewGroup?,
10         savedInstanceState: Bundle?
11     ): View {
12         binding = AlignrFragmentBinding.inflate(inflater, container,
13             ↵ false)
14         binding.lifecycleOwner = viewLifecycleOwner
15         binding.alignrViewModel = alignrViewModel
16         return binding.root
17     }
18
19     override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
20         super.onViewCreated(view, savedInstanceState)
21         viewLifecycleOwner.lifecycleScope.launch {
22             alignrViewModel.start()
23         }
24     }

```

Exemplary AlignrFragment. The fragment initializes and provides data binding to update values in a View.

Listing 4.3

sensor readings, a common and immutable data structure is generated that corresponds to a 3-dimensional vector $\langle x, y, z \rangle$ that includes a timestamp t . For identification purposes, the sensor type ϕ is also stored along. Therefore, we can define a sensor reading as a measurement of the following shape $\vec{x} = \langle \phi, t, x, y, z \rangle$. The prepared sensor values are forwarded to the `EVENTBUS` directly after receipt and are accessible by all other components at this point in a read-only manner.

FixedFrequencyCollector

A related component called `FIXEDFREQUENCYCOLLECTOR` is used to aggregate sensor readings in a composite measurement with a fixed frequency denoted as f . This addresses the fact that multiple sensors yield data with differ-

ent delays; however, different calculations within the framework depend on an integer flow of data. Thus, the `FIXEDFREQUENCYCOLLECTOR` subscribes to specific events on the event bus, namely all types from the `ANDROIDSSENSORSOURCE`. It creates time-based bins in terms of time delays to resolve the target frequency f . Intermediate events from the event bus are collected and averaged by type. In particular, readings from a sensor $\vec{x}_{t'}$ at time t' are integrated into a single measurement for time t by averaging values $\{\vec{x}_{t'} \mid t - f^{-1} \leq t' \leq t\}$ grouped by type using `mean`. The output of this component is an event itself. However, this event has a special type that provides the averages as a new \vec{x} for each type ϕ identified as `AlignrSensor.TYPE_FIXED_FREQUENCY_MEAN`. Averaging sensor values also address random bias in the sensor readings as the impact of this error is reduced. Consequently, this component performs sensor fusion across time [130] but does not execute any data verification methods such as sanity checking.

4.5.2 Middleware

Next, the middleware is a central layer of the architecture, as it implements the observer pattern to ultimately link components. In addition, it provides the *Single Source of Truth*.

StateMachine

First, a `STATEMACHINE` keeps track of the current state of *alignr* for multiple properties, each having its set of values (such as a 3-dimensional vector or enum value) and the corresponding timestamps. This is particularly important for keeping track of events. For instance, the `STATEMACHINE` memorizes the beginning of a standing phase and its ending. It is crucially important that all components perform calculations in the same state; hence updating states within the `STATEMACHINE` is a sensitive process limited to specific components, while reading is not restricted. The `STATEMACHINE` use Kotlin's `MutableSharedFlows` and `MutableStateFlows`. `MutableSharedFlows` also keep a history of state changes eventually needed for speed calculation. Next to the current sensor readings (including derived values) and limited history, the `STATEMACHINE` keeps track of the following information:

- ▶ A state defining a vehicle is currently moving or not (as reported by the `STANDINGPHASEFINDER`)
- ▶ An assessment of whether the vehicle is currently in a curve (as reported by the `CURVEEVENT`).

- ▶ The current speed as determined by the framework taking into account all factors (as derived by the `SPEED` component).
- ▶ The time of the last detected and applied event (as determined by `BUMPEVENT`, `CURVEEVENT`, or `STANDINGEVENT`).

EventBus

A core component of the framework is the `EVENTBUS` as it bridges multiple loosely coupled components in a many-to-many relationship. Its purpose is to provide access to all events generated through the framework globally. An event is defined as any structured output of a component comprising a type ϕ , a timestamp t and a payload such as a 3-dimensional vector or a state value as explained previously. This component is implemented as a *Singleton* and dependency-injected via Hilt into each component in the same way as the `STATEMACHINE`. The `EVENTBUS` may be considered a First-in-First-Out queue, where the oldest events are dropped if the queue is running full. Furthermore, all events share the same queue to streamline the workflow as components may be interested in multiple events at the same time. This way, they only need to subscribe to one event channel, eventually reducing overhead. This component is also built on top of Kotlin's `MutableSharedFlow` similar to the `STATEMACHINE`.

Mirror

The `Mirror` component listens on the `EVENTBUS` to detect relevant information. For selected cases, it may update the `STATEMACHINE` accordingly.

Calculation Layer - Events

4.5.3

The subclass of events of the calculation layer is intended to implement the event detection introduced in Section 4.3.2. Consequently, it contains detectors for road defects (simplified as bumps), curves, and standing. We define a confidence level for each event in order to be able to make a statement about how reliable a speed determination based on the event is. However, events do not yield a speed but rather reference points to estimate it without relying on an integrative approach.

BumpEvent

The detection of bumps or road defects is a multistep process based on CEP (c.f. Sidebar C) that takes into account the restricted computational capacity of the mobile environment in contrast to the stringent requirements that continuous autocorrelation places on the system.

Finding peaks based on acc_z

First, relevant points of interest, i.e. peaks, must be identified in the continuous stream of data from acc_z . To address bias and noise in the sensor data, we keep a running standard deviation \widetilde{std} based on a sliding window of length ω_B and overlap $o_B = \frac{1}{f}$. Once \widetilde{std} is above a threshold $\theta_{B,\sigma}$, this reading is considered a potential bump that can be matched, as this observation is inevitable once the vehicle passes a road defect. However, a peak p is defined as a sequence of exceptional std values, hence, a single outlier is immediately discarded. Consequently, we store for a peak p : $p = \langle t_s, t_m, t_e, std \rangle$. The timestamps define the beginning of the peak (i.e. $\widetilde{std} > \theta_{B,\sigma}$), the time where \widetilde{std} shows the highest readings (i.e. the “peak” of the peak), the end time (i.e. $\widetilde{std} \leq \theta_{B,\sigma}$), and ultimately the peak’s highest \widetilde{std} reading.

Fast matching peaks

Once two or more peaks have been captured in the data stream, they are checked in an adjacent step to see if they match the pattern that would occur if a vehicle were to pass over a road defect. We denote the set of peaks to be examined as $\mathcal{P} = [p_1, p_2, \dots]$. Based on this set, we form the Cartesian product $\mathcal{P}\mathcal{P}$ (yielding a set and peak pairs of the form (p_a, p_b)) and filter the resulting peak pairs accordingly so that the following holds: $\forall x \in \mathcal{P} \times \mathcal{P} : x_a \neq x_b \wedge x_{a,t_e} < x_{b,t_s}$. One would anticipate that the first deflection measured on the front axle would also be evident in the measured data when the rear axle passes. However, it cannot be assumed that two identical series of measurements will result since the location of the smartphone and other circumstances tend to counteract this [393]. Therefore, the next stage is to check whether a peak pair is valid based on straightforward and fast-calculable metrics. Indeed, this is the case if the distance between two peaks and the respective peak maxima should be within a certain window. Subsequently, the shape of the peak course can be determined. Here, the times already identified can be used, where the transition from the peak of the front axis to the peak of the rear axis occurs at the point where the minimum value of \widetilde{std}' in the period p_{i,t_s} to p_{j,t_s} is found for a pair of peak (p_a, p_b) (the time point is called $t_{\widetilde{std}'}$). Since the peaks should be courses of equal length (assumption of constant velocity while passing a pair of peaks), symmetrically arranged around the peak, the period of observation is $p_{a,t_s} = p_{a,t_m} - (p_{b,t_s} - t_{\widetilde{std}'})$ and vice versa for the ending time of p_b . Peaks are discarded from the set of peaks \mathcal{P} once they become obsolete.

Autocorrelation for speed estimation

Next, autocorrelation is performed on meaningful pairs of peaks according to Yu et al. [417]. This includes an in-depth comparison of the course of acc_z readings of both peaks p_a and p_b from a pair. Consequently, this autocorrelation is done to verify if p_b is a shifted (delayed) version of p_a . However, since the exact position of the peaks is not known, it is indispensable to analyze the lagged version as

initially explained. Let dur be a utility function to calculate the duration or length of a time series such as a peak p , mean a method to calculate the mean of a series of values, and std is the already known method to derive the standard deviation of a series. In summary, we calculate

$$\mathcal{R}_p(\theta_{B,l}) = \frac{1}{\text{dur}(p_b)} \cdot \frac{E \left[(p_b - \text{mean}(p_b)) (p_{b+\theta_{B,l}} - \text{mean}(p_{b+\theta_{B,l}})) \right]}{\text{std}(p_b) \cdot \text{std}(p_{b+\theta_{B,l}})}$$

For convenience, calling a function as defined above on a peak equals executing it on the acc_z series that are found in the time span of that peak. Recall that acc_z is derived from an accelerometer event that has a timestamp attached; hence the selection is straightforward. We use a normalized version of \mathcal{R}_p based on the peak duration. \mathcal{R}_p is symmetric for lags, i.e. $\mathcal{R}_p(\theta_{B,l}) = \mathcal{R}_p(-\theta_{B,l})$ holds [417]. Lags are evaluated up to a maximum of $\tau_{B,l}$. The lag with the highest confidence is then used to calculate the distance between both that represents the passings of the front and rear axle of the vehicle. Together with knowledge about the wheelbase of the car, the speed can be calculated. The yielded score \mathcal{R}_p is analyzed subsequently to guarantee that the peak pair is no false positive, as this will ultimately corrupt any estimation. In fact, *alignr* keeps the state of the last speed estimation with high confidence (such as a standing phase) and allows the new speed guess to be in a certain margin; otherwise, it is discarded. To further optimize the True Positive (TP) and reduce the False Negative (FN) rate, we trained and derived a logistic regression model that takes into account different metrics such as the peak duration, correlation, or uniqueness of the sensor's readings. The confidence of the prediction scales with the correlation of the two peaks as it is within $[-1, 1]$. The whole bump event is encapsulated in a composed event `AlignrSensor.TYPE_COMPOSED_EVENT` that is submitted to the `EVENTBUS`.

CurveEvent

A snapshot of the current speed may also be estimated using curves, as these are not dependent on the integrative values of the accelerometer and thus are not susceptible to global error propagation. Hence, the `CURVEEVENT` component watches the gyroscope to trigger appropriate calculations.

Basically, curves are detected using the gyroscope, with the z-axis being of particular interest. In order to process curves, the smartphone-to-vehicle alignment process must first be completed so that gyroscope events (`Sensor.TYPE_GYROSCOPE`) can be rotated appropriately so that gyr_z can be interpreted in terms of the vehicle. Other axes are irrelevant and only pose random noise.

Curve
detection

A curve is present once the gyroscope readings (or, more precisely, a running average to account for noise) exceed a threshold $\theta_{C,gyr}$. It continues as long as the running average is below a second threshold that increases with the length of the curve to mimic the natural driving behavior.

Impact of
curve length

A curve is only considered a valid curve if its length is sufficiently long, say $\theta_{A,r}$ readings above the threshold. Thus, the natural driving behavior is taken into account. Even if, in principle, the calculation of the speed is possible continuously on the basis of the centrifugal forces, this is only for *clear* curves and provides useful results.

This method updates the curve state within the `STATEMACHINE` and also creates a composed event `AlignrSensor.TYPE_COMPOSED_EVENT` with a curve payload.

StandingEvent

The detection of standing phases is performed based on a change of the appropriate state stored in the `STATEMACHINE`. The confidence of this event is absolute, i.e. 1, as the standing phase assessment is reliable but may be delayed a few milliseconds; however, this is irrelevant for major speed deviations. This value is updated by another component, namely the `STANDINGPHASEFINDER`. While this component only analyzes the sensor stream for changes that imply a change in the standing state, `STANDINGEVENT` also memorizes the duration and time of change. A suitable composed event `AlignrSensor.TYPE_COMPOSED_EVENT` with stance phase information is fed according to the `EVENTBUS`.

4.5.4 Calculation Layer - Finders

The main purpose of finders is to identify specific patterns in the sensor data. Compared to events, elements identified by finders are not composite events that fuse different sensor data.

AccelerationFinder

The `ACCELERATIONFINDER` is responsible for providing a cleaned acceleration sensor event that is based on the accelerometer readings as provided by the accelerometer but cleaned from any gravity. The currently valid gravity vector is provided by the `STATEMACHINE`. In order to preserve the separation of functionality, no rotation of the readings is performed within this component as no information about the current orientation is known. Furthermore, to reduce

noise, the `ACCELERATIONFINDER` keeps a long-running standard deviation of standing acceleration vectors to reduce potential noise in the cleaned acceleration vector. Eventually a corresponding event of type `AlignrSensor.TYPE_ACCELEROMETER_NO_GRAVITY` is submitted to the `EVENTBUS`.

ForwardVectorFinder

Apart from the z-axis that can be estimated based on gravity, the y-axis of the car can be determined by observing the acceleration vector while no steering is present. Recall that both axes are crucial for finding the smartphones orientation within the car (see Section 4.3.3).

Hence, the `FORWARDVECTORFINDER` observes the linear acceleration vector⁹ (that provides a sufficient enough quality for this task while still being infeasible for speed estimation as explained previously) and the gyroscope. In order to find a straight acceleration, `FORWARDVECTORFINDER` keeps rolling averages of both the linear acceleration vector and gyroscope vector. Once recent readings of the former vector are exceeding a given threshold called $\theta_{C,l}$ while the latter is below another threshold denoted as $\theta_{C,gyr}$ it is assumed that the car is driving in a straight line without any centripetal forces being present as no steering maneuver is performed. Eventually, all readings from the linear acceleration (that is cleared from gravity by Android with the explained drawbacks) roughly correspond to the vehicle's y-axis. The distribution of the vector can then be used to find the unit vector \vec{j} (subsequently also called direction vector). This process is continuously performed and updates the unit vector once requirements are met.

Derivation
process

The direction vector is protected from flipping by sanitizing the linear acceleration readings, i.e. only minor updates to the unit vector are permitted. At this point, it should be mentioned that the linear acceleration as it is derived here cannot make any statement about the direction. In particular, it is unknown whether the acceleration occurred when driving backward or forward. However, fuzzy logic within the speed calculation will take this uncertainty into account. The calculated and updated forward vector is presented to the `EVENTBUS` as `AlignrSensor.TYPE_FORWARD_VECTOR`.

Direction-
(un)aware-
ness

GravityFinder

In order to gain information about \vec{k} , i.e. the unit vector that corresponds to the vehicle's z-axis, the `GRAVITYFINDER` analyzes the gyr_z stream. As a

⁹`Sensor.TYPE_LINEAR_ACCELERATION`

precondition, this component requires a known “vehicle movement state” as it needs to gain knowledge about gravity while standing.

Gravity
rotation

Each reading from the gyroscope is used to rotate a gravity vector that is derived during a standing phase where only gravity is impacting the device. The actual rotation is done as explained in Section 4.3.4 using quaternions generated based on the gyroscope. The resulting pure quaternion can be considered as a new gravity that was rotated according to the driving profile. Due to the brief time intervals between the rotation steps performed by this approach, a little rounding error may be introduced with each measurement due to the restricted decimal places.

Rounding
error com-
pensation

As this rounding error accumulates, the features of quaternion multiplication might begin to distort the length of the gravity vector. To address this problem, the method also checks the length of the gravity vector and scales it appropriately if a variation is discovered, ensuring that it always keeps a similar length as gravity can be considered roughly constant within the boundaries of a trip.

Drift com-
pensation

The gyroscope, whose data is the basis of the rotation, is susceptible to a small but constant drift, which accumulates over time and causes an erroneous calculation. To counteract this error, the angle between a low-pass gravity vector (from the standing periods) and the current, rotated gravity vector is first calculated (briefly referred to as α). This is based on the assumption that the standing phase-based vector is close to the ground truth vector direction and therefore reflects the correct angle of the gravity vector. The current, rotated gravity vector is rotated again by α using the Rodrigues-rotation formula [91]. Depending on the current acceleration, this angle can be adjusted since a correlation between drift and acceleration vector could be identified empirically.

The resulting rotation is submitted to the `EVENTBUS` as a new sensor type called `AlignrSensor.TYPE_ROTATED_GRAVITY`.

StandingPhaseFinder

The `STANDINGPHASEFINDER` collects `SENSOR.TYPE_ACCELEROMETER` readings to preserve a rolling standard deviation of this vector’s length to detect periods with low magnitudes. However, it only selects readings within a 99.9% confidence interval around the standard deviation readings to exclude readings posing harsh changes in accelerometer readings, probably due to measurement errors. Once the resulting standard deviation is below a threshold $\theta_{S,\sigma}$ for minimal period of time of $\theta_{S,l}$, a standing phase event of type `AlignrSensor`.

TYPE_MOVEMENT_STATE is crafted, submitted to the EVENTBUS and the STATEMACHINE is updated accordingly or vice versa.

Integration Layer

4.5.5

Finally, the different information flows together in the integration layer to estimate a speed.

VehicleCoordinateAlignment

As soon as the unit vectors \vec{j} (as provided by the FORWARDVECTORFINDER) and \vec{k} are known, coordinate rotation can be performed, and ultimately, smartphone-to-vehicle alignment is achieved. Hence, this component listens for `Sensor.TYPE_GRAVITY` and `AlignrSensor.TYPE_FORWARD_VECTOR` events to continuously update $\vec{i} = \vec{k} \times \vec{j}$. After an initial calculation of \vec{i} , new measurements (e.g. acceleration vectors) can be rotated, so they equal the vehicle's coordinate system as presented in Section 4.3.3. The rotation of any meaningful sensor reading is offered as a synchronous utility method; thus, the EVENTBUS is not involved. It should be noted that gravity is not removed prior to any rotation.

Speed

The SPEED component is a mediator to include all the information from the framework and is the core component to derive a speed estimate. It collects multiple events (i.e. `AlignrSensor.TYPE_ACCELEROMETER_NO_GRAVITY` and `AlignrSensor.TYPE_COMPOSED_EVENT`) and observes all states, namely, movement state and curve state. Four different submodules are implemented in this component that may estimate the speed using multiple but different information. The SPEED module performs hybrid sensor fusion according to the guesses \mathcal{G} provided by each submodule. Each guess g contains a confidence value that indicates how plausible the respective predictor believes its velocity estimate to be. Confidence can vary between $[0, 1]$. Once a submodule determines an estimate, it sends $g = \langle t, v, \chi_g \rangle$ to an aggregator ($\mathcal{G} \leftarrow g$) that applies multiple fusion algorithms to find the best guess of the current speed. In particular, probabilistic methods are used as well as the Dempster-Shafer decision theory [206].

A trivial but stable velocity source is the standing phase, which can be detected with high confidence. At the same time, this is the phase whose speed is precisely

Standing
phase

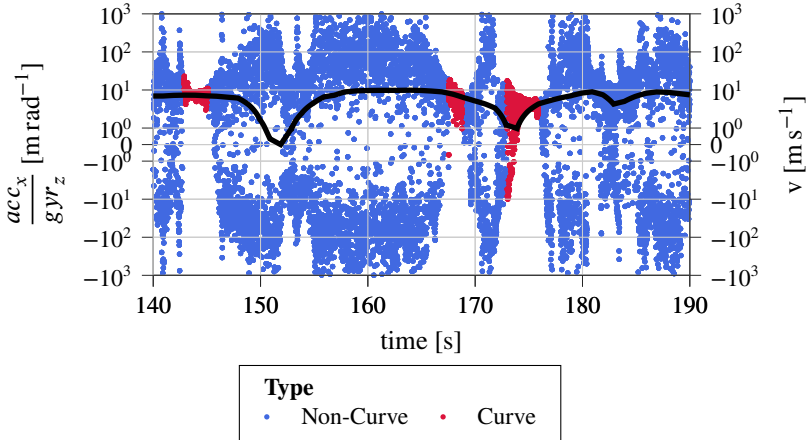


Figure 4.10 Relationship between the lateral acceleration (i.e. centrifugal forces) when passing through a turn and the rotation around the z-axis. Values are recorded by the gyroscope. The figure shows the supposed velocity $v = \frac{acc_x}{w}$ for one trip that matches the ground truth speed at *some* point in a curve. It is evident that there is strong and noisy fluctuation apart from curves.

defined so that $\chi_g = 1$ can be assumed. Thus, this speed source overwrites other sources, leading to an immediate setting of the speed to 0 m s^{-1} .

Integrative
approach

Given a current timestamp t_i , the integrative approach is based solely on the acceleration along the y-axis measured by the accelerometer acc_i and the time difference from the previous measurement t_{i-1} . The measurement values used of type `AlignrSensor.TYPE_ACCELEROMETER_NO_GRAVITY` require a rotation by `VEHICLECOORDINATEALIGNMENT`; otherwise, no exact mapping of the vehicle acceleration can be made. Thus, a new velocity can be derived using $v_{t_i} = v_{t_{i-1}} + acc \Delta \mathcal{T}$. *alignr* is a sampling-based approach; therefore, it is assumed that acc_i is constant for $\Delta \mathcal{T}$, which is approximately true for high frequencies. χ_g is chosen carefully for *gs* but static as no reliable source can be found.

Bump
evaluation

Once a bump event is found, it is processed by the subcomponent that extracts the speed along with the autocorrelation value. The autocorrelation value, which indicates the similarity of two peaks, is used as confidence because more similar peaks are more likely to be induced by the same road defect. Recall those bump events are only triggered once `BUMPEVENT` detects such an event, and hence false positives are not to be handled by `SPEED`.

Curves are continuously evaluated once the respective state is present in the STATEMACHINE.

Curve
following

Curve progression as an influencing factor. However, curves have to be carefully handled, as, in theory, they should yield the correct velocity, but this assumption does not hold for the whole phase due to overlapping noise and bias. This becomes evident w.r.t. Figure 4.10 showing the relationship of lateral acceleration and rotation around the z-axis as recorded by the gyroscope. Apart from the curves that are highlighted accordingly, the result is random noise and is not feasible for estimation. The fluctuation is significantly reduced when entering a curve phase. Now, the calculation should yield the vehicle's current speed, although this is not the case, especially at the beginning, when results are blurry. In fact, the speed estimation is correct at some point of the curve as the GPS recorded speed (considered ground truth) crosses the result of $\frac{acc_x}{gyr_z}$. It is challenging to identify this corresponding intersection point in the progress of the curve. As is trivial to see, a point can be found in the advanced course of the curve, yet it is not constant.

Intermediate Speed Determination. It should be noted that the point to be found is the one at which the fluctuation of the calculations is reduced to the minimum, as this allows for a sufficiently accurate estimate based on the curve event. Therefore, based on this finding, the first $\tau_{C,d}$ values of a turn are ignored for evaluation. Then, for a window of length ω_C , the mean values $a\bar{c}_x$ and $g\bar{y}r_z$ are formed, and based on this, $v = \frac{a\bar{c}_x}{g\bar{y}r_z}$. If the rolling standard deviation $std(v)$ (window size ω_C) is found under a 1 m s^{-1} , the estimate can be used.

Finalizing a curve guess g . For a final guess g , two parameters are still undefined. First, the confidence, which is formed based on the curve length $\Delta\mathcal{T}_C$ and an exponential function $(1 - \exp(\frac{\Delta\mathcal{T}_C}{-2.5}))$, which aims to favor later values again and finally serve as a more accurate estimate compared to the integral method. Second, the final velocity is to be found based on a low-pass filter ($\alpha = 0.8$) to incorporate a new estimate more as a look back at Figure 4.10 reveals that the fluctuations may indicate the intersection point.

Based on the obtained \mathcal{G} , the aggregator must select the guess $g^* \in \mathcal{G}$ that has the highest plausibility and can serve as a basis for a velocity estimate at time t_i . It is possible that the set \mathcal{G} has contradictory elements, and, in turn, these conflicts must be resolved. For example, a standing phase may be detected, but \mathcal{G} s are still provided by the integrative approach, although it is known that the standing phase might be the truth.

Aggregation
of multiple
sources

Filtering the messages of modules. g^* is determined based on several selection criteria. However, first, this set has to be descendingly sorted: First, by χ_g , then by t . This drifts the process towards prioritizing more plausible guesses in favor of recency. The resulting multilevel ordered sequence (called \mathcal{G}^*) is then to be evaluated by the following predicates to determine a valid guess.

1. Let \check{g} be the current speed guess approved in the previous iteration. Select only those $g_j \in \mathcal{G}^*$ where $\text{sign}(g_{j,v}) \neq \text{sign}(\check{g})$. This is most likely related to a measurement error, and since a sudden direction change is reasonably unlikely, this guess has to be ruled out. A small margin is allowed to account for sign changes at particularly slow speeds as they occur after standing.
2. Remove all speed guesses from \mathcal{G}^* that occur after a standing phase whose speed exceeds a threshold $\theta_{v,j}$. Guesses above that threshold are considered an unrealistic jump start.
3. Find the most recent speed guess $\hat{g} = \text{argmax}_{g \in \mathcal{G}^*}(g_t)$ and then ensure that $\forall g_j \in \mathcal{G}^* : g_{j,t} - \hat{g}_t < \theta_{v,t}$ holds. This eliminates all guesses that are too old and therefore do not reflect the *current* speed of the vehicle. At the same time, this also prevents the SPEED module from picking a guess multiple times, producing unrealistic constant plateaus.

Selecting the best guess. Consequently, the first element of that list is the new $g^* = \mathcal{G}^*_1$ as it has a high confidence, while being reasonably recent after passing all filters. However, if $\mathcal{G}^* = \emptyset$, no speed estimation can be made and the current \check{g} is also valid at t_i .

Sanity checking event-based guesses. A fair amount of uncertainty is inherent in the use of events such as turns and bumps because, although velocity estimates are not affected by error propagation, deviations from the ground truth may nevertheless occur. So-called *reference points* are used to counteract this. Reference points are points in time when an event was last selected as g^* . Let t_{ref} be the last known and accepted reference point. A reference point can be used to determine the approximate speed frame using a function denoted as rfw in which a future speed guess may be located, provided it is based on a turn or bump. Therefore, speed estimates that would lead to a strong, potentially unrealistic divergence (i.e. false positive events) can be mitigated. The basis of this handling is that the prior estimate is close to the ground truth. The longer the previous reference point is away, the larger the confidence interval in which the value may lie, since simultaneously, with more extended trip progression, the uncertainty grows. The window is calculated using $\text{rfw} = (g_t - t_{ref}) * \delta v_{ref}$ where g is an

example speed guess and δv_{ref} is a constant that determines the window growth. It was empirically estimated to be 0.0005 m s^{-1} . t_{ref} is always set to the end of a standing phase, should it be finished with the necessary information being credibly provided by `STANDINGEVENT`.

Due to the blurred orientation of the device in the vehicle, no reliable statement can be made about the direction of travel (c.f. Section 4.5.4). For this reason, a sanity check is performed that continuously conducts a frequency analysis on the estimated velocities. It is a valid assumption that the majority of velocities in a trip are positive, which corresponds to a forward heading ride. If this is not the case, the direction vector is flipped once. This leads to subsequent velocity calculations assuming the correct direction of travel since no further flip can occur due to the calculation of the forward vector.

Speed sign

Persistence Layer

4.5.6

Repository

The persistence layer, with its repository component, is tasked with storing data and information that is produced and processed during a run. The repository has to meet special requirements in terms of processing speed since data is generated with high frequency, and backpressure must be explicitly avoided. Events of different types are persisted using the high-performance NoSQL database `objectbox`¹⁰, which is specifically targeted in mobile and resource-limited environments. Data is stored in batches to reduce system load since synchronous persistence, i.e. storing an event as soon as it appears on the `EVENTBUS`, is impractical from a performance point of view. In the repository, the data is organized by *runs* which serve as a container for speed guesses, event occurrences, and event readings. Runs are unique and immutable data structures that represent a single trip. Refer to Section 5.2 for a further explanation of trips. A run can be exported into the corresponding JSON objects.

Interface Layer

4.5.7

The Interface Layer provides access to all results of *alignr*. Refer to Section 4.4.2 for an explanation on how to access the data as *alignr* provides common support for Android interfaces.

¹⁰<https://objectbox.io/>

4.6 Evaluation

In the following, an evaluation of *alignr* is presented, which was conducted empirically.

Environment The evaluation was conducted under real conditions in daily traffic with a focus on urban areas. Different smartphones, arbitrarily placed in different vehicles, were used for the experiments. The reference value for speed is GPS, although it should be noted that this is also subject to a certain degree of uncertainty of approximately 3 km h^{-1} [417]. Selected experiments were performed under laboratory conditions, and this is appropriately noted here. Parameter values have been estimated based on exhaustive optimization methods or empirical analysis.

Structure We first analyze the data preparation steps, including gravity removal and coordinate alignment steps. Subsequently, we analyze the accuracy of the event detection and the validity of the speed estimation. Afterward, the speed estimation is discussed with the example of multiple trips and circumstances. Finally, we present performance insights of the implementation w.r.t. the constrained environment.

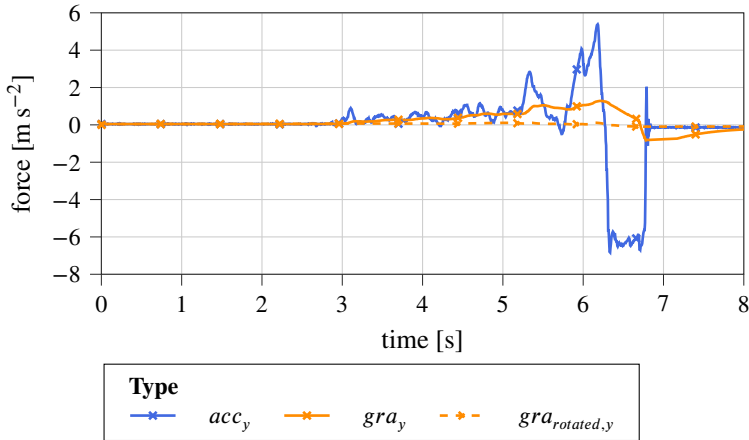
4.6.1 Gravity Removal

First, the efficiency of the gravity removal approach is illuminated as a precise gravity vector is essential for further processing.

Accuracy

We conducted different experiments to assess the accuracy of the gyroscope-based gravity rotation. This enables an in-depth understanding of gravity and its behavior under different circumstances, with the gravity assumed to be constant.

Overview of approaches As mentioned in Section 4.3.4, Android provides a `Sensor.TYPE_GRAVITY` that employs a low-pass filter to remove apparent accelerations from accelerometer readings. However, the quality of this method is not precise, and therefore it is infeasible to process sensor values in the context of *alignr*. The proposed solution uses a high-sampling gyroscope vector that enables one to rotate a gravity sample accurately recorded during a standing phase. Differences in terms of accuracy and robustness are depicted in Figure 4.11. This experiment was executed as follows. A device was placed flat on a plane surface and moved along the y-axis, resulting in linear acceleration. As expected, the gravity should not change and remain almost constant throughout the experiment without showing any



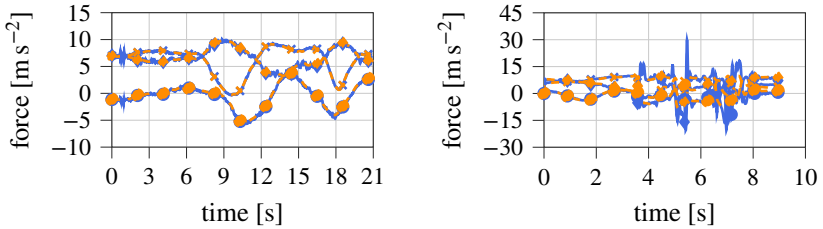
Example of the acceleration along the y-axis. It is obvious that the low-pass-filtering-based approach of Android for the determination of the gravity provides time-delayed values compared to the rotation of a stationary phase gravity. The more significant the measured acceleration is, the more the value provided by Android changes.

Figure 4.11

fluctuations because no orientation change occurs. The measurement proves that the low-pass filtered Android gravity (gra_y) roughly follows the course of the accelerometer readings (acc_y): Changes are more significant with high fluctuations of acc_y , but they are not constant as one would expect over the progression of the experiment. As a result, this result is in no way usable for speed determination. This is different from the proposed gyroscope-based approach that yields a gravity vector $gra_{rotated,y}$ of almost zero along the y-axis because the gyroscope does not collect rotation due to the design of the experiment.

To assess the quality of the gravity preprocessing, experiments were conducted without any linear acceleration that may falsify the gravity vector. Figure 4.12a shows the measurement record in which the devices were carefully moved with multiple degrees of freedom. One can see that the rotated gravity matches the raw accelerometer readings closely. Hence, the continuous integration of the gyroscope to rotate the gravity is usable in this case. However, a slight offset can be observed when rapid and harsh movements affect the device (c.f. Figure 4.12b). The impact is visible by the high fluctuations in the raw accelerometer readings where the gravity sample does not align anymore, ultimately resulting in displaced levels when entering a “standing phase” at the end of the experiment. As in the former example, one would expect that gravity axes and raw accelerometer

Impact of movement



a) Smooth rotation. The gyroscope-based rotation of a gravity sample matches the gravity as it is recorded by the accelerometer.

b) Harsh rotation. The rotation of the gravity is susceptible to error propagation induced by the gyroscope.

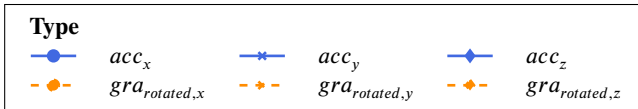


Figure 4.12 **Example of the impact of rotation on gravity.** A rotation without any acceleration of the recording device indicates that the gravity is also rotated along the axes and matches the reading of the accelerometer.

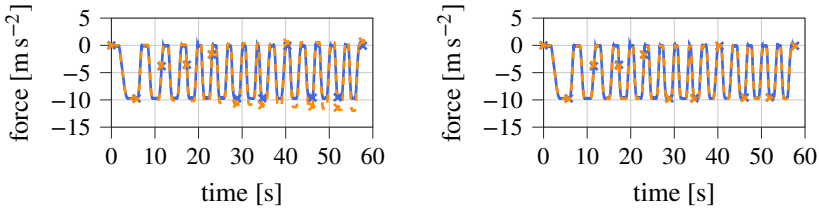
axes roughly match, but that is not the case. We can state that the gyroscope is also susceptible to drift and consequently introduces error propagation into the system under circumstances that may occur in real-world scenarios.

Drift Analysis

The previously identified drift has to be addressed to yield useful gravity samples. Figure 4.13a illustrates a gyroscope drift with the gyroscope vector increasing with time when periodic movement is applied to the device.

Error
propagation
of the
rotation

The drift results become apparent in the standing phases. It could be assumed that all standing phases output the same level of acceleration values since only gravity acts there. However, because of drift, a shift in gravity compared to the acc values by a certain, non-constant, or recurring delta can be recognized. Therefore, the rotation through the gyroscope is not exact. Consequently, this bias is also found in measurement data outside of standing phases but cannot be determined due to the complexity and degrees of freedom of the process. To reduce this problem, gravity is continuously updated when entering a standing phase, so that error propagation of the gyroscope rotation only has an influence between these phases.



a) Uncalibrated readings. Even with a constant motion, a drift of the gyroscope can be recognized, which leads to the fact that the rotation of the Gravity is more substantial with the increasing length of the analysis and grows in sum over the length of the actual gravity vector.

b) Compensated readings. Gyroscope drift compensation successfully prevents the rotated gravity vector from unrealistically lengthening throughout the measurement.



Example of drift acting on the gyroscope. The drift leads to error propagation of the gravity rotation. No additional acceleration is added to the system during the measurement.

Figure 4.13

Two additional optimizations were applied during gravity rotation, although a trade-off between too aggressive alteration and original data retention must be found. During rotation, rounding errors can unintentionally change the length of the gravity vector, which is unrealistic for short time steps (recall that the gyroscope is sampled with $f = 200$ Hz). Therefore, we rescale the new rotated gravity sample once the deviation becomes too notable to compensate for this change. A second optimization technique is to reintroduce orthogonality between all axes of the rotated gravity sample compared to the raw accelerometer reading (we use a moving average accelerometer reading to account for noise), as there is an angle drift. Although both readings should not align precisely, a rough overlap may be expected. A suitably small multiplier for this angle adjustment might ensure that, unlike the low-pass strategy, this correction does not end up overcompensating and sacrificing crucial linear acceleration. Similarly, when the car accelerates for an extended period, the raw acceleration measured deviates from true gravity. The correction method described herein begins by eliminating the angle difference between the two vectors, thereby bringing the gravity vector closer to the real one, including linear acceleration. This is the purpose of the

Further optimizations

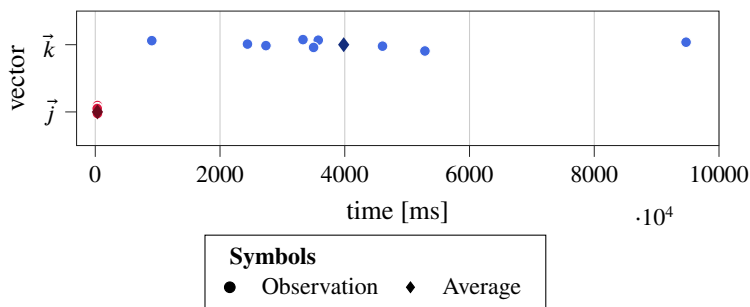


Figure 4.14 Illustration of the delay to provide coordinate alignment. The delay before the vectors needed for the coordinate alignment are obtained varies depending on the type, yet both can be determined at the beginning of a trip. While the gravity vector \vec{k} seems to be available immediately in all cases, the detection of the direction vector \vec{j} takes longer depending on the trip. This is due to the necessity to find a forward acceleration without steering movements.

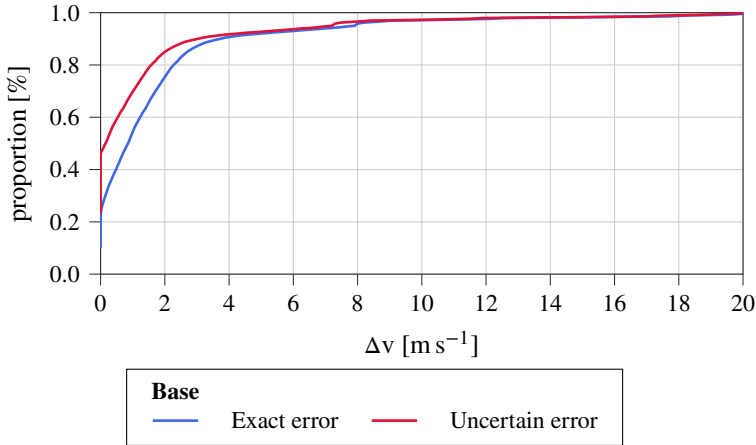
Rodrigues-rotation formula mentioned above with a sufficiently small angle correction of 0.05 % for each measurement, eventually rotating the gyroscope vector towards the reference vector. Although this removes small portions of the linear acceleration, the overall gravity accuracy can be increased. Figure 4.13b illustrates that gravity and accelerometer readings for one axis align, proving that drift is successfully eliminated.

4.6.2 Coordinate Alignment

For an applicable smartphone-to-vehicle alignment, the stable but early determination of the two vectors \vec{j} and \vec{k} is required. The latter is directly determinable in standing phases, whereas the former requires specific driving maneuvers. Thus, the delay in determining both vectors is of particular interest.

Gravity vector If \vec{k} is unknown, the underlying gravity can be detected at each stance phase of the vehicle. Since this is usually the initial state at the beginning of the recording of a trip, this vector can be provided almost immediately. This state is also confirmed by the measurements (c.f. Figure 4.14).

Direction vector In contrast, more complex preconditions must be met to determine the direction vector \vec{j} (c.f. Section 4.3.3). On average, this takes around 4000 ms for \vec{j} to become available after starting the vehicle (c.f. Figure 4.14). Since both vectors



ECDF plot depicting the delta between the ground truth (i.e. GPS) and the *alignr*-based speed values. The plot also shows the case if the inaccuracy of the GPS-based speed is also considered. It is obvious that for latter case the more than 80 % of all speed values have no deviation at all proving that IMU-based speed estimation is feasible.

Figure 4.15

are collected continuously, *alignr* was optimized to provide the coordinate system as early as possible, although possibly less accurately.

Accuracy of Speed Estimation

4.6.3

Next and most importantly, the speed estimation accuracy is presented, and the impact of events is discussed. On this basis, the feasibility of the guessed speed values for distance estimation is described.

First, the general deviation of speed guesses w.r.t. the ground truth is of particular interest. Figure 4.15 presents an ECDF plot that illustrates the distribution of the deflection. Shown are differences from the exact speed of GPS. One can see that it is below 1 m s^{-1} for around 50 % of all measurements, with around $1/3$ of all estimates having no errors. 80 % of all measurements still have a deviation of approximately 2 m s^{-1} . This analysis assumes a correct GPS speed, which is not realistic and is too strict an assumption. Therefore, we also plotted the ECDF plot when the ground truth was assumed to be blurry. Thus, we added an error band based on the findings of Yu et al. [417] that takes into account the uncertainty of the estimates of the GPS (i.e. $\pm 0.7 \text{ m s}^{-1}$). As a result, the precision of *alignr*

Summary
insights

can improve to a deviation of almost 0 m s^{-1} when considering 50 % of all values once this new ground truth is used.

Table 4.5 Descriptive statistics of all trips used for the evaluation.

#	Δv_{Q25} [m s ⁻¹] [†]	Δv_{Q75} [m s ⁻¹] [‡]	mean(Δv) [m s ⁻¹]	max(Δv) [m s ⁻¹]	$\sum t$ [s]	$\sum d$ [m]
1	0.0	1.4	0.59	3.83	408	2537
2	0.0	2.07	1.06	4.91	175	1305
3	-0.14	0.96	0.25	5.12	333	1305
4	-0.0	7.35	3.18	21.75	419	1637
5	-0.08	1.03	0.31	4.45	456	2390
6	0.0	1.88	0.9	5.23	272	1642
7	-0.08	1.2	0.48	4.67	335	1967
8	0.0	1.43	0.47	4.31	162	863
9	0.0	1.21	0.47	4.48	294	1467
	-0.03	2.06	0.86	6.53	2854	15113

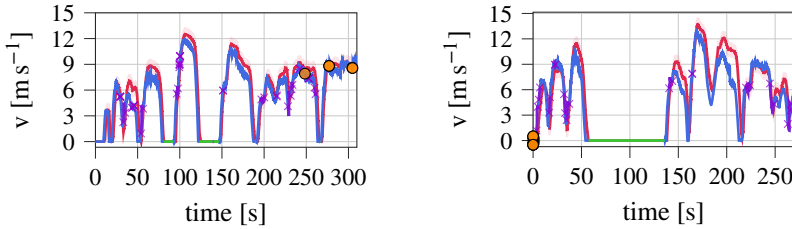
[†] Denotes the .25-quantile for the series [‡] Denotes the .75-quantile for the series

In-depth
results

Next, we analyzed all test tracks in detail. In total, these trips were 47 min long and covered around 15 km of mostly urban roads. The results are listed in Table 4.5. The table shows the .25 and .75-quantiles for the speed delta between both data sources calculated for every single measurement. In addition, the mean error is shown as well as the maximum deviation. The average mean error is as low as 0.86 m s^{-1} . *alignr* in some cases predicts lower speeds compared to GPS as the .25-quantile shows. 75 % of all values have a deviation up to 2 m s^{-1} , which was already made clear by the ECDF plot. However, when looking at individual trips, one can see that one trip is heavily biasing the results as trip #5 posts a deviation of over 7 m s^{-1} for the .75-quantile. This is probably mainly related to the coordinate rotation process that errored at some point, eventually degrading all predictions until the next standing phase.

Example
trips

Figure 4.16 presents two example trips. Both trips illustrate that *alignr* is able to estimate the velocity within or close to the range of the ground truth. Also, one can see that events support the correction of a drifted inference process.



a) **Example A.** The trip contains multiple curve and bump events and long movement phases.

b) **Example B.** The trip has an extended standing phase and some events.



Comparison of the speed estimation of *alignr* and the ground truth speed of the GPS. Figure 4.16
Included is the inaccuracy error band. Events are highlighted.

Benefits of Events

Events have proven themselves to considerably support speed estimation, as confirmed by our experiments and related work [417]. However, their support differs depending on various characteristics. For instance, curves must be considerable, i.e. their length and homogeneity must meet the requirements introduced in the previous section.

Descriptive statistics illustrating the impact of events for speed estimation. The more events are used, the better the prediction. Table 4.6

Feature	Δv_{Q25} [m s ⁻¹] [†]	Δv_{Q75} [m s ⁻¹] [‡]	mean(Δv) [m s ⁻¹]	max(Δv) [m s ⁻¹]	Improvement [%]
+ integral	-0.21	1.07	0.47	5.47	—
+ curves	-0.12	1.0	0.42	4.45	-11.33
+ bumps	-0.34	0.55	0.11	4.19	-74.77

[†] Denotes the .25-quantile for the series [‡] Denotes the .75-quantile for the series

Table 4.6 depicts the feasibility of estimating the speed of different trips. Events were added consecutively to increase the number of reference points available Accuracy increment

for exploitation. The table again shows the .25- and .75-quantiles as well as the mean and maximum speed delta between IMU and GPS speed. Taking a look solely at the integral-based approach, which includes standing phases, reveals that the mean error is about 0.47 m s^{-1} . As this alone does not sound significant, the .75-quartile shows a deviation of 1.07 m s^{-1} indicating that most estimations are off. Adding curves as reference points does not significantly improve the quality of the prediction. This may be related to the uncertainty when using the curve speed, as it is challenging to select the correct point in the progression of a turn. Furthermore, the prediction of speed within a curve is also susceptible to various disturbance variables, such as the vehicle's sway or the slipping of the smartphone. Hence, improvement is limited. However, this is different for bumps since an estimation on their basis depends exclusively on the correlation selection of two related peaks, and no other dynamic or hard-to-achieve assumptions are necessary. This is directly reflected in a sharp decrease in the average speed difference, which drops about 75 % to 0.11 m s^{-1} . In particular, the .75-quantile is reduced to 0.55 m s^{-1} ultimately delivering accurate speed predictions in most cases. Therefore, the quality of the estimate is comparable to the entire evaluation data set discussed in the previous section and shown in Figure 4.15.

Correlation Between Events and the Mean Error

We further analyzed whether there is a correlation between the mean error and the number of events detected on a trip. Basically, this assumption can be confirmed, although the strength of the correlation varies.

Standing
phase

If standing phases are considered, a positive correlation of 0.42 is found. Interestingly, this means that more standing phases lead to a higher deviation between GPS-based and IMU-based velocity. At this point, two reasons can be given. First, the IMU can record finer movements, whereas Android's GPS is reduced for energy reasons in terms of frequency when there is no slight movement, and, in addition, due to its resolution, small differences in speed are not apparent. With frequent standing phases, there is still the condition that the speed changes significantly, for example, due to strong acceleration and braking maneuvers. Due to the limited sampling of the GPS signal, the quality of the ground truth is also questionable, and the differences, even with smoothed GPS-based speed values, between both sensor speeds become blurred as uncertainty in the data set increases. At this point, it is worth mentioning again the significant discrepancy in GPS speed under some conditions [417].

Curves and
bumps

The correlation is strongly negative for the other two events, with values of -0.77 for turns and -0.65 for bumps. As more of these events are encountered,

the velocity deviation from the ground truth decreases. The significance of the tests may be limited due to the unequal number of turns (on average 16 per trip) and bumps (on average 51 per trip), but a sufficient impression is obtained. The velocity determination is more accurate with bumps since a direct calculation is possible independent of other measured values, such as previous speed guesses. Additionally, unlike curves, no safety parameters have to be observed, which means that fewer of these events are discarded. Finally, it can be confirmed that more frequent events contribute considerably to the quality of speed determination.

Distance Derivation

Throughout this work, the speed collected using the IMU will be used in different cases and scenarios to derive the distance, for example, enabling privacy-invasive attacks such as the one presented in Chapter 12.

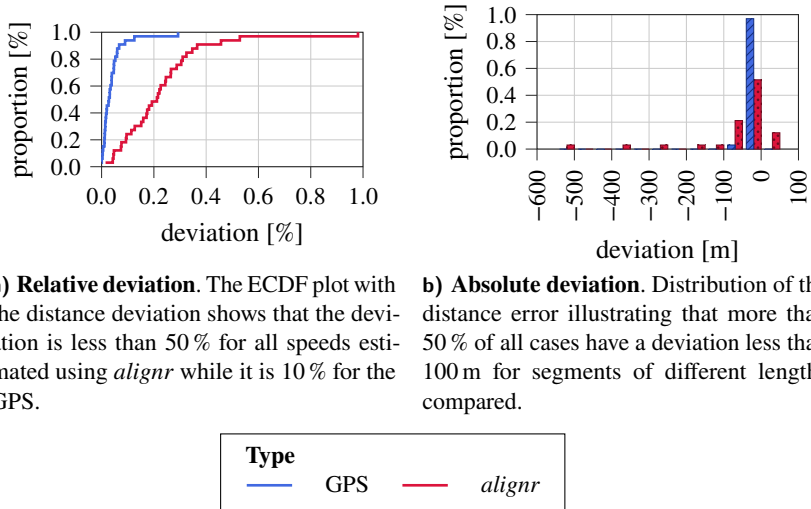
In the following, we will compare the feasibility of both sensors, namely GPS and IMU for this task. The first one uses the geodesic distance between two adjacent location coordinates, while the latter one uses the integral of the speed. Both results are compared with the ground truth, which is the shortest-path distance as reported by OpenStreetMap (c.f. Sections 5.5 and 12.6.2).

Figure 4.17a illustrates the accuracy using an ECDF plot. It shows that both distances employ an error, yet the GPS-based distance is more accurate compared to the IMU-based one. However, we can state that even though the GPS distance has a lower deflection, the derived distances are still sufficiently accurate in both cases. This becomes apparent when the absolute deviation of the distance deviation is analyzed. In Figure 4.17b the distance error compared to the ground truth is categorized in 50 m bins. It shows that around 50 % of all readings have a deviation of up to 50 m for IMU-based calculations while this value is around 95 % for the other sensor. However, most importantly, distance errors above 100 m are rarely observed. This may be a crucial threshold when thinking about a road network with intersections and the task of relative localization based solely on distances (c.f. Chapter 12). The exemplary task requires the accuracy of a prediction below the distance of consecutive intersections.

Performance Statistics

4.6.4

Finally, we respect the mobile yet highly constrained environment of *alignr* by providing performance insights.



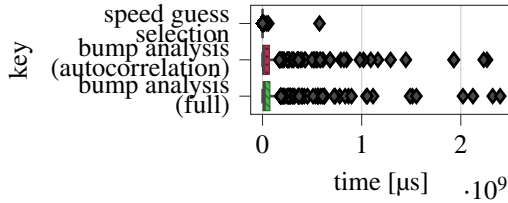
a) Relative deviation. The ECDF plot with the distance deviation shows that the deviation is less than 50% for all speeds estimated using *alignr* while it is 10% for the GPS.

b) Absolute deviation. Distribution of the distance error illustrating that more than 50% of all cases have a deviation less than 100 m for segments of different lengths compared.

Figure 4.17 Illustration of the distances derived from *alignr* and GPS. Readings are compared to the ground truth extracted from OSM show an expected deviation.

Accuracy of
frequency

First, the accuracy of the `FIXEDFREQUENCYCOLLECTOR` is of interest. A stable flow of measurements is vital as most calculations rely on the time delta. For instance, gyroscope-based gravity rotation is a core functionality of the framework, as many other operations rely on it. We claimed that the rotation between two successive gyroscope readings might be approximated as a straight line while the time intervals are small enough. Therefore, having too slow computations of any kind when creating `AlignrSensor`. `TYPE_FIXED_FREQUENCY_MEAN` may lead to unexpected behavior. Our experiments were carried out with a target frequency of $f = 200$ Hz that results in an intended period of 5 ms between two successive measurements. It is shown that the mean delay between two measurements is as accurate as 5.0018 ms with a standard deviation of 0.0042 ms. The minimum was found to be 4.0000 ms, and the maximum observed delay was 5.0115 ms. The `FIXEDFREQUENCYCOLLECTOR` is implemented as a Kotlin flow using the native time-based sampling method `sample`. As such, the accuracy is defined by the OS, but we ensured that no additional overhead or heavy computations might interfere with the OS scheduler.



Selected performance statistics for crucial operations of *alignr*. Shown are the duration of the speed selection process and two metrics for the road defect analysis.

Figure 4.18

Next, we illustrate some of the essential computations that are worth mentioning in this context. It should be noted that some operations are based on the successive collection of sensor data (e.g. curve evaluation); therefore, they are not directly assessable. Figure 4.18 shows boxplots for three selected operations, namely the speed guess decision-making process, the full run-time of the bump evaluation, and the run-time of the autocorrelation in this context. The first is relevant because the decision process occurs with a frequency that is equal to the collection frequency (i.e. $f = 200$ Hz). It is well beyond a critical threshold and shows an almost immediate speed estimate. For the other two metrics, one can state that the autocorrelation process is time-consuming as it is the major contributor to the bump detection process.

Run-time of calculations

Conclusion, Error Sources, and Outlook

4.7

In this chapter, we considered the possibility of having the speed of a vehicle estimated by a smartphone located in the vehicle. With *alignr*, we present a PoC application that allows the assessment of the speed based on the IMU data without the need for GPS with a viable accuracy in several settings.

Overview of Literature

4.7.1

First, a SLR was performed to find existing methods that perform the velocity estimation task. During the analysis, it was found that most of the works are based on the accelerometer, followed by the gyroscope. However, it is not common for approaches to be appropriately publicly available, let alone suitable for being embedded in existing applications.

Several failure reasons are found in the literature that coincide with the typical sensor problems addressed in Section 2.3. The impact on velocity detection is

Deflections and counter-measures

significant, with a broad consensus among the works. The literature suggests a wide variety of approaches that can be used to reduce the impact of the mentioned errors. Among them are, in addition to the initial orientation of the device (recall smartphone-to-vehicle alignment), the use of smoothed sensor values, and also the use of other reference points as proposed by Yu et al. [417]. It is shown that ex-post approaches that compute the velocity for each time point after the completion of a trip are superior to real-time approaches in terms of accuracy.

Taxonomy Building on the results, a taxonomy is presented that divides the methods for the final determination into five superordinate classes. Pattern-based approaches, in which the addressed events (reference points) are to be found, are to be mentioned. Furthermore, calibration-based approaches present tools to achieve smartphone-to-vehicle alignment, for example. Approximation-based approaches are mainly dominated by sampling methods due to the nature of sensor data acquisition. Next, filter-based approaches include low-/high-pass filters or sophisticated Bayesian filters. Last, AI-based methods are expected to learn the relation between speed and sensor data on their own.

4.7.2 Our Approach

The developed taxonomy provides a useful overview of which methods are suitable for speed estimation. Consequently, we integrate the different approaches into a holistic concept within the framework of *alignr*. In contrast to related work, where most approaches are not retrofittable, *alignr* was designed to work as an Android module that supports state-of-the-art data binding via known interfaces. Moreover, the framework does not require extended learning or setup phases, such as controlled movement of the device. It is designed for the mobile environment, reducing its performance and energy footprint.

Smartphone-to-vehicle alignment *alignr* first uses calibration methods to perform smartphone-to-vehicle alignment to allow for arbitrary positions within a vehicle. The process puts certain requirements on the setting, although it was shown that these are met within the first seconds of a trip.

Gravity removal Also, *alignr* assesses gravity as the most significant, roughly constant bias. Through experiments, it has been shown that the Android-provided functionality to separate linear acceleration and gravity from an accelerometer does not yield sufficient accuracy. Thus, we propose a gyroscope-based method that constantly rotates a gravity sample that is collected while standing. As the maintained gravity vector depends on the previous one, error propagation may occur, especially in vivid movement settings, but is optimized within the framework.

Apart from integrating the speed from gravity-cleansed accelerometer readings, the application also uses pattern-based approaches together with filter- and approximation-based methods to find reference points, which are eventually used to provide a speed estimate independent of previous measurements. Therefore, the reference points are robust against error propagation. We define three reference points based on related work [417], namely standing phases, road defects, and curves. These are summarized together with the integrated speed to provide a reliable speed guess.

Events

Extended experiments and data analysis reveal an accurate speed estimation compared to a ground truth derived from GPS, although this sensor itself is susceptible to inaccuracies. Real-world experiments prove *alignr*'s ability to precisely predict the speed with an average deviation from ground truth as low as 0.86 m s^{-1} for our test data set. Furthermore, 50 % of all values have no offset at all. In contrast to the integration-based approach, adding events as a speed source reveals that the speed error can be reduced around 75 %. This relationship is also confirmed by a strong correlation of -0.77 for curves and -0.65 for bumps. Additionally, more events generally yield better results. More importantly, *alignr* will be used within this work to e.g. enable self-localization based on distances. Distance estimation (which in turn is based on velocity, but errors have a slightly lower impact) has been proven to provide satisfactory accuracy, with half of all readings within an interval of $[-50 \text{ m}, 50 \text{ m}]$.

Results

Error Sources

4.7.3

We now briefly mention the identified errors that were encountered during the development and optimization of *alignr*.

Standing phases are identified based on various assumptions, which are defined by parameters explained in the following. Depending on the smartphone used, the mounting position in a vehicle, or the vehicle itself, the measured values differ during standing phases. Hence, we observed that no static solution or parameter set is feasible to detect standing phases across devices with the best accuracy. Remarkably, the variance of the accelerometer readings can vary significantly in the standing phase. An initial calibration may support accurate detection of the standing phase.

Immediate standing phase

Apart from the exact progression step when selecting the current centripetal force and the gyroscope reading during a curve to perform a speed guess, they are therefore indisputably a source of error. They are complex driving maneuvers in which a wide and fast-changing variety of forces and rotations

Curves

act simultaneously on the system consisting of the vehicle and the smartphone. During the development, it was found that gravity rotation is often particularly error-prone at these points. One idea to compensate is to use backtracking to correct the gravity vector once a reference point is found, although performing backtracking is challenging in real-time scenarios. At this point, the present reference point is a curve; thus, this velocity must be used accordingly to estimate the present gravity starting from the acceleration vector. However, this approach is only limited fruitful since it assumes that an exact velocity of the turn can be extracted, which in principle must not be the case due to the described problems.

4.7.4 Outlook

For future work, it is possible to implement backtracking approaches for possible error sources, but the real-time requirement still has to be considered. Although *alignr* performs continuous calibration for multiple instances, such as standing phases, gravity, and orientation, the calibration preset has to be optimized. In particular, *alignr* should be extended to detect external sources of errors, such as phone sliding e.g. due to improper fixation within a car. Such scenarios are currently not handled at all and will only be corrected by the remaining error correction methods that are, for instance, performed during standing phases. It also makes sense to use AI-based methods to detect bumps. Currently, these are confirmed using a static regression model.

The approaches presented in the rest of the dissertation (see Chapters 8, 11, 12 and 14) often process Floating Phone Data. Hence data of different shapes is of interest. A central data pool should enable empirical evaluation of the proposals and provide potential comparability between the different scenarios.

This chapter is structured as follows. Requirements for a data set are introduced in Section 5.1. The collection process and the applied notation throughout are presented in Section 5.2. A presentation of the data follows in Section 5.3. Next, “*events*” are introduced that describe specific situations in a data set (c.f. Section 5.4). Finally, we introduce in Section 5.5 the OpenStreetMap which serves as a map resource within this work.

Structure

Requirements

5.1

A data set is a collection of data with the data coming from sensors that have been introduced in Section 2.1. Since sensors can retrieve data of varying quality and granularity, and each use case may pose several ambitious requirements, the collection process must ensure to satisfy the objective of providing a holistic data set. Thus, we define three requirements that are briefly described in the following:

Real-world data The data set should be able to provide real insight into everyday driving situations that are determined by external factors. First, driving situations are defined by the geographic area and the underlying road network. In particular, the data set should contain data from urban and rural environments as well as different types of roads, such as residential or highways. Furthermore, constantly alternating traffic situations impact the data collection process, as traffic varies depending on the time of day, weather, and day of the week, which should be accounted for accordingly. At this point, overlaps of different factors are desired.

Diversity In addition to external factors, internal factors must be considered, including each driver’s driving habits and reactions to various

traffic situations, to generate a diverse data set. To obtain a diverse representation of driving behavior, different participants with different backgrounds (age, distance per year, driving experience) should collect data without instructions. Data collection will be carried out independently of the vehicle or device. Drivers shall drive in their own vehicle or rental vehicles on the one hand. There should be at least one data collection device in the vehicle, placed either arbitrarily or controlled. This can be used to check various factors that influence sensor data and, if necessary, evaluate them.

Use case agnostic The data will support the evaluation of various concepts within the scope of this work. As this thesis relies on sensor data and location-based information, the data set should contain as much information as the sensors provide. In selected cases, the data set may contain an over-representation of certain driving events if this is necessary to investigate a situation. Generally, the results should correspond to the frequencies found in the traffic system.

5.2 Collection

We now explain the process of collecting sensor data that will generate the document corpus for this work. In addition, a fundamental notation is introduced that is applied throughout the different chapters.

5.2.1 Process

Data collection was carried out using the PoC application (c.f. Chapter 4) or ROADR (c.f. Chapter 8) application over a period of four years. More than 15 people participated in the study using 16 different devices as listed in Table 5.1. Therefore, the study ensures varying sensor accuracy [219] and settings due to the wide range of mobile devices with different ages and price ranges. Multiple work [319, 324, 326] and [S1, S2, S11, S12] contributed to the data set. Further, different vehicles from multiple manufacturers and price ranges are included. Participants either used their own car or shared it among others. Driving experiences range from a few years to multiple decades, ensuring a diverse pattern in driver behaviors.

Overview of mobile devices SP used to create the data set. The devices were running different versions of Android. Also some devices are part of a device pool.

Table 5.1

Device	Operation System	Pool
Google Pixel 4a	Android 11	
Oneplus 6	Android 10	
Samsung Galaxy A3	Android 8	2
Samsung Galaxy A71	Android 10	
Samsung Galaxy Note 10+	Android 11	
Samsung Galaxy S6	Android 7	1
Samsung Galaxy S7	Android 8	
Samsung Galaxy S8	Android 9	
Samsung Galaxy S9 - 1	Android 9	
Samsung Galaxy S9 - 2	Android 10	
Samsung Galaxy S9 - 3	Android 10	
Sony Xperia XZ Premium	Android 8	1, 2
Sony Xperia XZ2	Android 8	2
Sony Xperia XZ2 Compact	Android 10	
Xiaomi Mi Mix 2	Android 9	
Xiaomi Redmi Note 4	Android 6	

To create the data set, the proponents \mathcal{P} were equipped with a smartphone sp that runs an instance of one of the two applications. They were instructed to place the device in their vehicle in a steady position to avoid sliding during the trip. However, no additional requirements were made regarding the characteristics of the route. This yields a wide range of trajectories that match the requirements introduced in Section 5.1:

Requirements fulfillment

- ▶ The traveled routes cover realistic scenarios, such as traveling between residential areas, from country roads to the city center, workplaces, shopping stores, schools, and universities. However, it should be noted that due to the nature of the reason for collecting the data, some ingrained and specialized routes were also recorded, such as roundabout passings or road work passing. They do not represent a typical trip.
- ▶ Due to the number of participants, the data set is diverse in terms of driving style and reactions to various traffic circumstances to allow

further data analysis and allow generalization. Some journeys are recordings of everyday lives, i.e. commuting, and reoccurring routes.

- ▶ Trips incorporate different times to account for different traffic conditions throughout the day.
- ▶ Next, different areas were covered, including urban and rural areas as well as a multitude of different types of roads, although urban areas were primarily focused.

5.2.2 Notation

Sidebar B Series Notation

This work often uses a series of values due to the nature of sensor data. The following symbols are used to denote specific types of data series in this work.

$\{a, b, c, d, \dots\}$ is called a **set** of elements without any order, although the elements are distinct and can only be part of that set at most once. It has no specific size and can also be empty.

(a_1, a_2, \dots, a_n) is an ordered, fixed-size collection of n elements of the same type called **vector**. Each element can be referred to by its position (index) starting at 1. Note that we use subscripted indexes, e.g. $S = (x, y, z)$ and S_1 refers to x . Furthermore, we use the notation $[k]$ to represent the set (a_1, a_2, \dots, a_k) with k being the index $1, 2, \dots, k$. It cannot be empty.

$[b_1, b_2, \dots, b_m]$ denotes a **sequence** or a list that is an ordered collection of elements of the same type of variable length (here m). A sequence with $m = 0$ is called an empty sequence. An element is referred to by its index, similarly to a vector.

$\langle a_1, b_2, \dots, c_o \rangle$ is a **tuple** and describes a ordered list of elements of fixed size o but of potentially different type (also called a o -tuple). Elements of a tuple are addressed by their name; consider a tuple $T = \langle x, y, z \rangle$ with T_x referring to x .

In the course of a journey, sensor data is continuously persisted by the application on the smartphone in the form of measurements \mathcal{M} . A measurement m is a six-tuple and is defined as follows:

$$m = \langle t, \vec{acc}, \vec{gyr}, v, loc, h \rangle$$

with acc being the sensor readings from accelerometer, gyr denotes readings from the gyroscope and v is the velocity in m s^{-1} as derived by the GPS. Furthermore, location information (loc) may be available as well as the vehicles current heading h as derived by the magnetometer (mag). Both can be NULL, i.e. no sensor value is available (e.g. there is no external GPS signal or the reading is intentionally not recorded). Hence we define $loc = \{\emptyset, loc\}$ and $h = (\emptyset, h)$, respectively. We further define a location as $loc = (lat, lon)$ with the coordinate system being WGS 84 (EPSG:4326)¹.

Each reading from acc and gyr is a three-dimensional vector $(x, y, z)^T \in \mathbb{R}^3$ as defined in Section 2.1.2. Strictly speaking, it is therefore a tuple according to definition (c.f. Sidebar B), but the term vector is more common in this context. x_i denotes the i -th dimension of some sensor reading \vec{x} . For instance, acc_x represents the current sensor value in x direction from the accelerometer.

Shape of a measurement

We call a sequence of data points m a time series \mathcal{M} . Consider $\mathcal{M} = [m_1, \dots, m_u]$ to be a sequence with u measurements. Think of \mathcal{M} as a matrix with rows representing the u element long sequence and columns representing a measurement's values:

Formation of a time series

$$\mathcal{M} = \begin{bmatrix} t_1 & \vec{acc}_1 & \vec{gyr}_1 & v_1 & loc_1 & h_1 \\ t_2 & \vec{acc}_2 & \vec{gyr}_2 & v_2 & loc_2 & h_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ t_u & \vec{acc}_u & \vec{gyr}_u & v_u & loc_u & h_u \end{bmatrix}$$

We stick to the well-known list notation that $|\mathcal{M}|$ returns the number of elements in a list, i.e. the number of rows u . In addition, if a sequence \mathcal{M} can be considered a function, the inverse of that function returns for any $m_i \in \mathcal{M}$ the position $0 < i \leq u$ of that measurement in the sequence. As a utility, we call that inverse function $\text{idx} : \mathcal{S} \rightarrow \mathbb{N}$ with \mathcal{S} being any type of ordered structure.

Since \mathcal{M} are (time series) sequences, elements are ordered in time order, and in particular, a sequence is totally ordered. Each m provides a t for that purpose.

Order of measurements

¹ <https://epsg.io/4326>

Hence, elements in \mathcal{M} are indexed in monotone increasing order as $\mathcal{M}_{1,u;1}$ with $\mathcal{M}_{1,u;1} = (t_1, \dots, t_y)^\top \in \mathbb{R}^y$ representing a vector of all discrete timestamps (the first column) extracted from the measurements (see Chapter 2).

Measure-
ment slicing

$\mathcal{M}_{1,u;j,v}$ is called the slice of a matrix (similar to slicing in Python) and in this case yields a $(u - i) \times (v - j)$ matrix with only values scalars falling into the selected row and column range (i, u and j, v respectively). In our case, \mathcal{M} has the shape $u \times 6$ in every case and each row is a single m that is in turn a tuple. Elements in \mathcal{M} are addressed by the name, i.e. a name sequence $names = [t, \vec{acc}, \vec{gyr}, v, loc, h]$ exists. We therefore define $\mathcal{M}_{1,u;j,v} = (\mathcal{M}_{u,names_x})$ to represent a $u \times 1$ matrix (or column vector) of column $names_x$, e.g. t ($j = v = 1$). Hence, it is the initial example $\mathcal{M}_{1,u;1}$ that holds all timestamps $(t_1, \dots, t_y)^\top$. We also write $\mathcal{M}_{\bullet,names_x}$ if we slice column $names_x$ over all rows u . Note that $\forall z_l \in (\mathcal{M}_{u,t}) : z_l < z_{l+1}$ holds.

5.2.3 Time-related Properties

In general, data points for a time series are taken at successive equally spaced points in time, called sampling period, to project a continuous signal into discrete measures (c.f. Chapter 2). This results in a sampling rate denoted as frequency f that is the reciprocal of the sampling period. It is trivial to see that discrete measurements with higher f will better describe continuous signals. On the contrary, higher f are more challenging in terms of gathering, processing, and persisting. In addition, the frequency is limited by the underlying sensor (see Section 2.1.2). Our data set contains \mathcal{M} s with different frequencies, although the frequency within each \mathcal{M} was static to allow further data processing depending on the intended task.

Frequencies
in the data
set

In fact, with a target frequency of 30 Hz, the mean was 29.2 Hz with a standard deviation of 1.3 Hz. Setting $f = 25$ Hz, the mean was at 24.6 Hz and standard deviation decreased to 0.7 Hz. There were also recordings with a frequency as high as 200 Hz which was precisely the mean and a standard deviation lower than 0.1 Hz. We refer to Chapter 4 for more background on how to achieve a precise frequency.

5.2.4 Sensor Fusion

A measurement m according to the definition, inherits from one mobile device but by integrating multiple sensors into one reading. Thus different shaped

data is fused into one single representation. This process is called *sensor data fusion* [161, 210].

Fusion can be performed to get more reliable and accurate information. Forbes and Boudjemaa [130] define different dimensions on how to fuse sensor data. First, *sensor fusion* as is can be used if multiple sensors measure the same dimension. This addresses errors such as outliers or stuck-at-zero. Next, sensor fusion can be performed on *multiple attributes* to generate a more holistic description of a phenomenon. Third, fusion *across domains* can be used to generate a broader picture of a specific circumstance or attribute. For example, sensors at multiple instances can collect average speeds by deriving a macroscopic impression. Last, fusion *across time* can be used to generate more accurate sensor readings by e.g. combining recent recordings with calibrated ones to address drift or noise.

Introduction
to sensor
fusion

Recalling our definition of m , we apply multiple attribute sensor fusion where we integrate readings from the accelerometer acc , gyroscope gyr , and magnetometer mag as well as the GPS. Hence, we collect data in a cooperative sensor fusion scenario [118]. Also, we fuse across times to identify specific situations in the sensor stream, such as standing phases. Such special occasions are introduced in Section 5.4.

Application
of sensor
fusion

Additional Remarks

5.2.5

\mathcal{M} s are assigned to a tuple of driver and mobile device $\langle p, sp \rangle$. Figure 5.1 presents the frequency of mobile devices SP used in the collection process to generate the data set.

An exemplary extract of the recorded measurements \mathcal{M} that illustrates the shape of the sensor data is shown in Listing 5.1.

Presentation

5.3

In the following, we will present the collected data set. In total, 15 participants record 155 trips using 16 mobile devices. The cumulative trajectory length is 1662 km. Figure 5.2 shows that around 80% of all trips fall into the range between 2 km and 6 km.

Area

5.3.1

The data is collected primarily in Regensburg, Germany, including surrounding areas. In some cases, data was also collected in the cities of Passau, Germany,

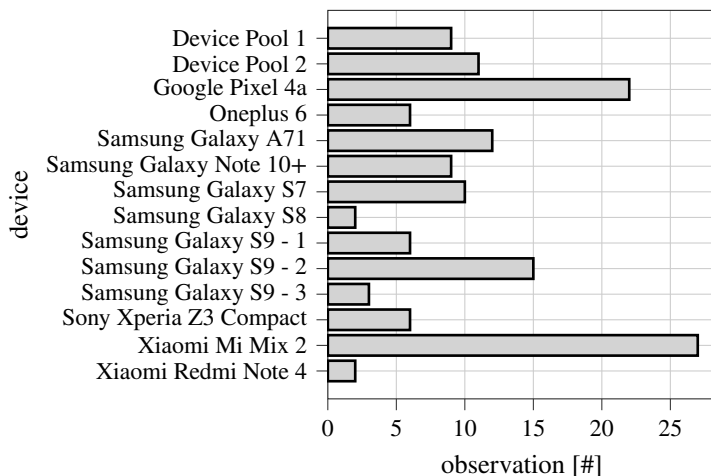


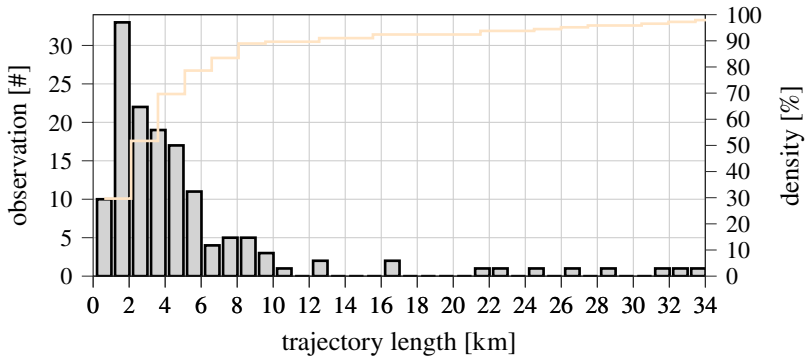
Figure 5.1 Distribution of trips recorded with a specific device $sp \in SP$.

```

1  [
2    {
3      "timestamp": 12312,
4      "acc": { "x":0.0023942748, "y":-0.481249243, "z":9.7782182693 },
5      "gyro": { "x":0.025785232, "y":-0.0051599788, "z":0.0083505278 },
6      "speed":8.2583837509,
7      "location": { "lat":48.9884970456, "lng":12.2092330476 }
8    }, {
9      "timestamp": 12347,
10     "acc": { "x":0.0766167939, "y":-0.4740664065, "z":9.5028762817 },
11     "gyro": { "x":0.0074592759, "y":-0.0131659787, "z":0.0107939895 },
12     "speed":8.2583837509,
13     "location": { "lat":48.9884970456, "lng":12.2092330476 }
14   },
15   ...
16 ]

```

Listing 5.1 Exemplary extract of recorded measurements \mathcal{M} illustrating the shape of the sensor data. Data is persisted as JSON or CSV respectively.



Binned distribution of the trajectory length. In total, 155 tracks were recorded. Most recorded tracks have a length below 5 km with some spanning up to 34 km.

Figure 5.2

and Salzburg, Austria. Due to the purpose of the study, a focus was placed on urban scenarios, which means that the share of urban routes predominates. Furthermore, with a few exceptions, no freeway journeys are included, as these turned out to be of little informative value after a scenario and application analysis. This circumstance was exacerbated by the fact that at the time of data collection, construction sites were taking place on the sections of Autobahn 3 around Regensburg. As it turned out, such data could not be used for the planned applications within Part III.

Figure 5.3 illustrates the routes collected around Regensburg, Germany (highlighted area). The color of the roads defines the type of road, as this is stored in the OpenStreetMap (OSM) data. One can see that Regensburg is surrounded by two highways (A3 and A93, respectively) to the south and west (blue). The overlay shows how often an area has been traversed across all routes. The gridsize corresponds to approx. 500 m × 500 m sections in longitudinal and latitudinal directions. It can be seen that a large part of the Regensburg-originating routes crosses the area near the University of Regensburg. More rural routes outside the marked urban area are less frequently found in the data set on purpose. The red hexagon has been traversed a total of 162 times.

Distribution of trajectories

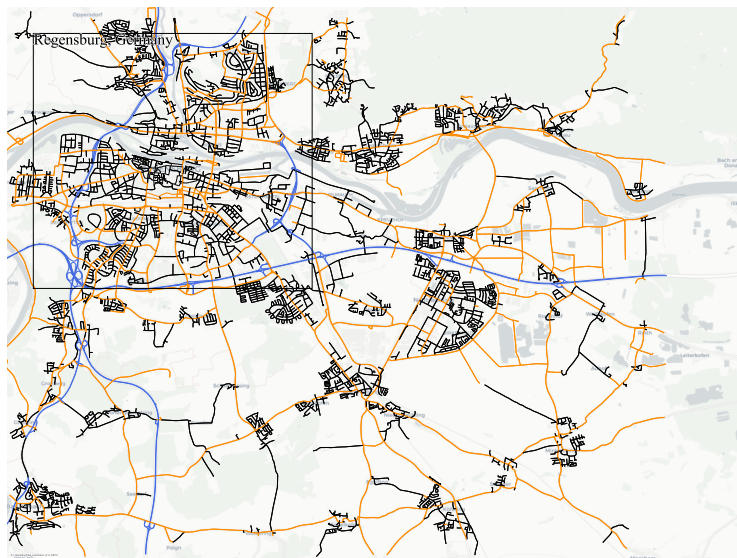


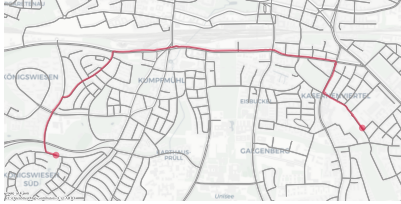
Figure 5.3 Heatmap of the covered within the data set. The map is rasterized into $\approx 500 \text{ m} \times 500 \text{ m}$ sections in longitudinal and latitudinal directions. There is a focus on urban areas, namely the city of Regensburg (highlighted area). Colors of the street represent the type of the connection.

5.3.2 Example routes

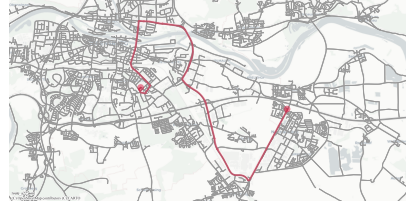
Figure 5.4 shows some selected representative routes. Highlighted in color are the route traversed and the underlying road network. The figure shows an urban trajectory (Figure 5.4a), a route that leads from the rural area to the city (Figure 5.4a), and a trip that is entirely in rural areas (Figure 5.4a), as well as a special recording without branches (Figure 5.4a).

5.4 Events

Hall and McMullen [161] define multisensor data fusion as “*the technology concerned with the combination of how to combine data from multiple (and possible diverse) sensors in order to make inferences about a physical event, activity, or situation*”. An event can, as a result of this, be defined as the change of the system’s state [168]. Sensors generate low-level events, i.e. sensor data.



a) Urban-focused route. The route runs exclusively in the city. Routes of this type often pass through traffic lights, which cause a change in traffic flow and thus represent an external influence factor. Likewise, lane changes take place on multi-lane roads. Vehicles of different types, such as cars, trucks, or buses, can be found and affect a driver's driving behavior.



b) Urban-to-rural transitional route. The route starts in a rural area and leads to the city. One can see a clear difference in speed between the two sections of the route also, the number of curves increases.



c) Rural-focused route. The passing of several small local sections with a reduction of speed to typical city-like speeds (e.g. 50 km h^{-1}) as well as the subsequent increase of speed to distant roads (e.g. 100 km h^{-1}) is gathered in this route.



d) Turnless route. A route with no turns may be uncommon and may not yield many events w.r.t. this work. Several places are passed on a straight route here.

Selection of exemplary routes. Some examples from the gathered data set show route-specific properties such as the passed area or trajectory shape.

Figure 5.4

Driving
behavior
dependencies

During a journey, the sensors continuously record data to conclude the vehicle's state or environment. Three different types of influence can be distinguished. First, influences can be caused by the behavior of the *vehicle driver*. This includes, for example, acceleration, which is reflected in accelerometer sensor readings, and the change from one road to another, i.e. a turning maneuver seen in the accelerometer and gyroscope readings. The latter is closely related to *road network*. This also affects the sensor data since the vehicle must move accordingly in the road network. The course of a road can also lead to changes in direction that the gyroscope can record. Finally, there are other *external factors* that have an impact on vehicle movement. These are not always clearly distinguishable from the other two categories, but the primary difference is their origin, intrinsic or extrinsic. Rush hour and associated lower speeds are considered extrinsic, as are evasive maneuvers or poor visibility. It is necessary to classify this circumstance accordingly. Chapter 11 considers this in more detail.

Sidebar C A primer on Complex Event Processing (CEP)

CEP is an approach to monitoring and analyzing processes or dynamic systems, i.e. system that follows a specific logic. For this primer, we refer to Hedtstück [168]. Based on this, additional processes or procedures can be set in motion. The basis of a decision is formed by events that occur in the original process and cause a sudden change in the state there. Events can be recognized *à posteriori* by means of a timestamp and a representative type. Events, according to Hedtstück [168], differ from activities, which elapse over some time and are bound by the start and end events. In particular, unlike events, activities do not change the system's state.

A CEP system structure events at different abstraction levels, horizontally or vertically. On a vertical basis, events are dependent on each other, meaning that they can form more complex types of events in combination or when aggregated. However, the partial events do not have to take place simultaneously. The combination of multiple sources, e.g. multiple sensors such as the GPS and IMU is called horizontal abstraction.

It is the task of CEP (or more precisely, a CEP engine) to detect events in a continuous stream of data, with some events masking others. In order to process the data stream, approaches such as windowing are applied. Common steps in a

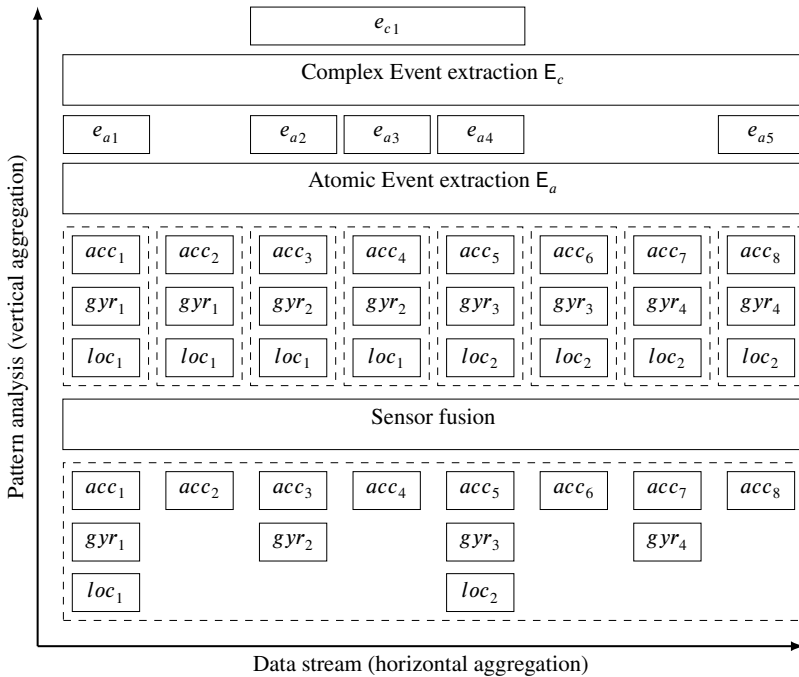


Figure 5.5

Relationship between Complex Event Processing and events. Horizontal aggregation focuses multiple sensor readings. Vertical aggregation then leads to longer patterns with atomic events e_a composed of successive measurement. Multiple e_a may be aggregated to more complex events \mathcal{E}_c .

CEP pipeline are filtering, preprocessing (which includes adding external data), pattern matching, and decision-making.

A common technology for processing fused sensor data to detect significant information is Complex Event Processing (CEP) (see Sidebar C). We apply vertical and horizontal CEP to find the sensor patterns of interest. Within this work, the following specific (complex) events and activities, respectively, are of interest:

Complex Event Processing

Acceleration as an *event* describes when the current moving state of a vehicle changes as a result of increasing speed.

Braking is the antonym of acceleration and indicates that the vehicle decreases the speed of movement (*event*).

Steering is an *event* that indicates a change in the vehicle's movement direction.

Turning is the *activity* change of lane or direction (e.g. turning on a road network).

Standing is the *activity* where the vehicle is not moving at all.

Driving is the antonym *activity* of standing.

The listed events are all complex events formed by atomic events that initiate the mentioned activities [237]. Figure 5.5 clarifies the relationship.

5.4.1 Extraction

Now we describe how the CEP is performed based on the available measurements \mathcal{M} .

Atomic events are generated by sensors in the form of sensor data and are denoted by e_a . An atomic event is always constructed by two adjecements, ordered measurements $(e_a) = (m_i, m_j)$ with both elements are part of a trip \mathcal{M}_i (horizontal abstraction). We define two different types of atomic events, namely *increasing* and *decreasing*. Let E_a a utility function that uses the difference $\Delta e_a = m_j - m_i$ to yield the atomic event type:

$$E_a(e_a) = \begin{cases} \Delta e_a < 0 & \text{decreasing} \\ \Delta e_a > 0 & \text{increasing} \\ \Delta e_a = 0 & \text{NULL} \end{cases}$$

e_a is not a valid event if E_a outputs NULL. However, this is unlikely because of the prevalent noise or bias of the respective sensors (i.e. accelerometer and gyroscope). A \mathcal{M}_i can contain an arbitrary number of events that are following their underlying *ms* order. Running E_a continuously for \mathcal{M}_i results in a sequence $\mathcal{E}_a = [e_{a1}, \dots, e_{a|\mathcal{M}_i|-1}]$.

Complex events are aggregated to craft the complex events *acceleration*, *braking*, or *steering* which in turn represent a sequence of measurements $[m_1, \dots, m_n] = \mathcal{M}_i$ (vertical abstraction). Consequently, for their identification, a sequence of events \mathcal{E}_a is analyzed to find ongoing sequences of the same event type, i.e. find

any $e_{ak} = e_{ak+1}$. Hence, we define another utility method $E_c : \mathcal{E}_a \rightarrow \mathcal{E}_c$. This method tries to greedily find a monotone² subsequence $\widetilde{\mathcal{E}}_a \subset \mathcal{E}_a$ that makes up a single, complex e_c . Complex events e_c s are partial sequences of \mathcal{M}_i , hence $\forall e_{ci} \in \mathcal{E}_c : e_{ci} \subset \mathcal{M}_i$. Depending on the given complex event type, different sensors are of interest. *Acceleration* and *braking* are obtained by analyzing the accelerometer, while *steering* events are derived based on the gyroscope. Complex events are then used to denote specific activities such as turning or standing³.

An activity is subsequence $a \subset \mathcal{M}$ of measurements $[m_i, \dots, m_j] = [m_l]_{l=i}^{i+K}$ with K items, with $m_l \in \mathcal{M}$ having start event i and end event $j = i + K$. Such sequences are found by i.a. analyzing complex events. Also, the sequence a has to meet certain conditions depending on the activity type.

Activities

- ▶ *Standing* and *driving* activities are extractable using data from the accelerometer. If no *acceleration* or *braking* event is present, one can assume a static situation of the vehicle. Section 4.3 illustrates an approach to detect the standing phases by analyzing the moving standard deviation of the accelerometer $\text{std}(acc)$. Phases which do not hold standing activity may in turn be considered driving activity.
- ▶ *Turn* activities are, in contrast, identified by waiting for steering events and watching the gyroscope. Turns are steering events that extend over a certain length and cause a certain change in angle. Given that the sum of the gyroscope readings for a steering event exceeds a given threshold, we consider that complex event to be a turning activity. A detailed introduction follows in Chapters 8 and 12.

Example

5.4.2

Based on the available sensor data, the real-time orientation of a vehicle or smartphone can be determined with the help of the gyroscope, enabling one to infer the historical trajectory. Furthermore, the vehicle's orientation and its respective changes can be set in relation to a global coordinate system. When using the magnetic north as a reference, one speaks of the vehicles *heading*. Heading is related to *bearing*. *Relative bearing* describes the angle between the

Heading and bearing

² Monotonic in the sense that successive events are of the same type.

³ Even more sophisticated information can be extracted by combining complex events, as this work will present in Chapter 8. The chapter introduces situations such as passing a traffic light, a roundabout, or roadworks and thus illustrates the potential of sensor data, framing this work.



Figure 5.6 Exemplary route including multiple turns and curvy parts. It starts at the lower right dot in urban area. Turns are marked with a blue cross, while the curvy parts of the road are marked green.

own heading and a reference object (e.g. destination in a road network). On the other hand, *magnetic bearing* is defined as the angle to magnetic north.

Turns,
curvature,
and straights

Figure 5.6 shows an example route. The highlighted elements (1), (3), (4), (6), and (9) (marked with a cross) denote turns as defined before. Furthermore, it can be seen that the trajectories between turns are also not a straight line but follow a particular shape (c.f. elements (2), (5), (7), and (8)). In the context of this work, the course of a trajectory is called slope, which can be qualitatively described as rather curvy or relatively straight (refer to Section 12.5 for a detailed analysis). The slope and curves can be recognized in the sensor data depicted in Figure 5.7. The relative heading change of the vehicle is drawn in since no reference to a reference coordinate system can be established via the gyroscope; each progression starts at zero. However, a global view is mandatory to determine the northward orientation of the vehicle. The summed measured values of the gyroscope⁴ are usable to detect further lane changes and the curvature of a track in addition to turns. Curves are characterized by a substantial change of direction, which corresponds to the definition mentioned previously. Furthermore, the direction of a turn can be recognized by evaluating the sign of the heading. In the case of straight sections, no differences can be recognized concerning the

⁴For reasons of simplicity, it is assumed that the smartphone is perfectly aligned with a vehicle, so that all gyr_z values are congruent with the change in the direction of the vehicle

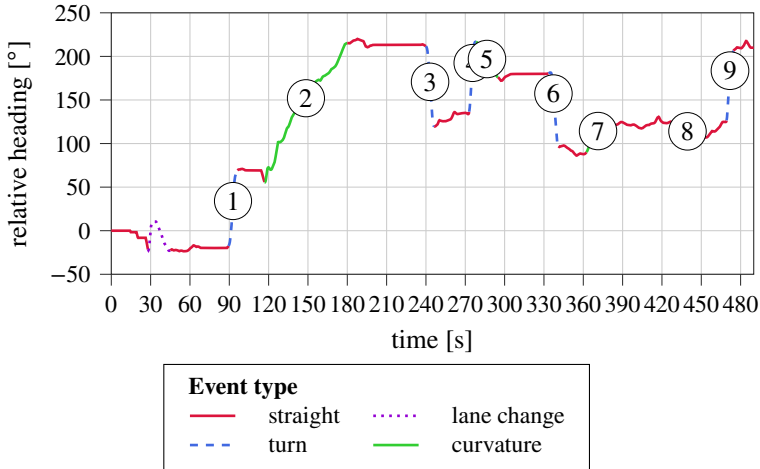


Illustration of the heading changes of the traveled route. Heading changes occur due to turns or road curvature. An exception is a lane change, that induces only a temporary heading change. As no reference system is available, each heading progression starts at zero.

Figure 5.7

heading so that it remains approximately constant. Curvature is an increasing change of direction that lies below the intensity of a curve but nevertheless leads successively to a deviation.

In the example mentioned above, an initial reference value of the heading is missing, which means that strictly speaking, one cannot refer to the heading. In this context, one could call it local heading, i.e. the own targeting measured from the start position. The value range of such a quantity is between $[-\infty^\circ, \infty^\circ]$ (if a vehicle would move infinitely in one direction, the cumulative changes of direction would also add up infinitely). To infer the actual heading, i.e. one's orientation w.r.t. the magnetic north, a smartphone's magnetometer can be used to provide an initial state. With this, the heading could be correspondingly expressed in $[0^\circ, 360^\circ]$.

Setting heading in relation

We performed experiments to assess the quality of the magnetometer, as high accurate readings are essential to yield meaningful and reliable heading references. Bias will be present; as we already discussed, the magnetometer is susceptible to sources of interference (c.f. Section 2.1.2). However, the evaluation should give insight into the significance of the error. The evaluation process was as

Magnetometer accuracy

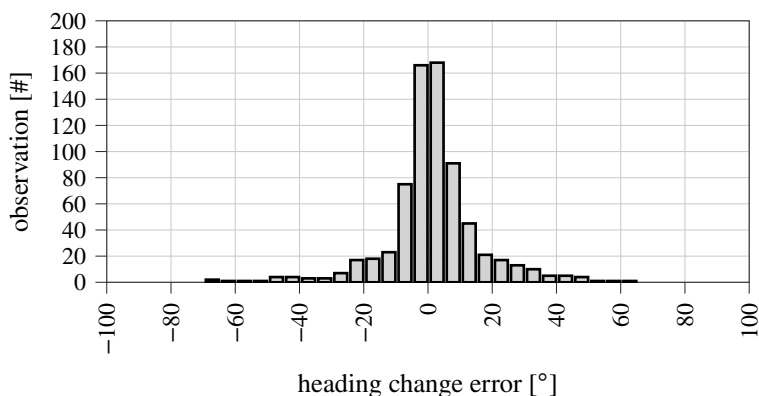


Figure 5.8 Distribution of the heading change error across multiple measurements. Around 40 % of all readings show an error around $\pm 5^\circ$.

follows. We compared the orientation as measured by the magnetometer with the magnetic bearing produced by OSM (see Section 5.5) and calculated the difference. In conclusion, as Figure 5.8 shows that there is a significant deviation depending on the vehicle, the smartphone, the position in the vehicle, the test area, or the orientation itself. Consequently, the magnetometer may only be used to roughly assess the orientation in a qualitative state, such as north, east, south, or west. We will cover this problem in Chapter 12. The results are in line with the results of related work [276].

5.5 OpenStreetMap

In addition to the motion information collected by an IMU, the work also evaluates and exploits location-based information (c.f. Chapters 7, 8 and 12). For effective processing of this spatial information, an appropriate resource is needed that provides detailed and accurate geo-information about road networks, including meta-information. Furthermore, an essential requirement for a map service is the ability to process available information efficiently.

Opensource
map data

Thus, another essential resource widely used in this work is OpenStreetMap (OSM), which provides extensive and comprehensive geospatial information. It includes next to a representation of the street network multiple additional attributes such as, for instance, traffic lights, lane directions, or speed limits.


```

1 <node id="4014860289" visible="true" version="1" changeset="37311670"
  ↪ timestamp="2016-02-19T16:11:16Z" user="****" uid="****"
  ↪ lat="49.0070014" lon="12.0867723">
2   <tag k="highway" v="traffic_signals"/>
3   <tag k="traffic_signals" v="signal"/>
4 </node>

```

Exemplary structure of a node. A node that represents a traffic light to control traffic flow on a highway.

Listing 5.2

OSM is an open-source project with voluntary contributors that globally manage and edit map data. The data is provided by OSM in an open data format without restrictions, eventually allowing customized geospatial information processing within the scope of this work.

Modeling

5.5.1

The data structure of OSM is built upon three basic elements, namely **nodes**, **ways**, and **relations** with additional information encoded as **tags** alongside. **Nodes** are used to define a geographic position, each defined by a pair of longitude and latitude coordinates (c.f. Listing 5.2). **Ways** are ordered sequences built from nodes that represent any given shape on the map by implying a connection between adjacent nodes. A connection between two nodes composes a way segment called a leg, which is the shortest line between those two nodes (c.f. Listing 5.3). Ways do not necessarily have to represent a road but can also be used to define any shape, such as buildings with shorter legs that model the real world more closely. A way may or may not be bilateral with a tag that indicates whether it is a one-way or two-way street in the context of a road network. Furthermore, a **relation** is an ordered list that can comprise nodes, ways, and relations, each of which can be assigned a role through a tag (c.f. Listing 5.4) The OSM syntax provides relations to define the logical or geographic relationships that exist between the listed components. Finally, **tags** are arbitrary key-value pairs associated with nodes, ways, or relations, enriching the map data with metadata⁵. Apart from basic characteristics such as the kind and the name of an element, tags hold information about qualities such as traffic lights, speed limits, or whether a way is a one-way street.

OSM data structure

⁵For a full list, refer to https://wiki.openstreetmap.org/w/index.php?title=Map_features&oldid=2111805

```

1 <way id="31104525" visible="true" version="22" changeset="113667776"
  ↳ timestamp="2021-11-11T20:26:51Z" user="****" uid="****">
2   <nd ref="662086518"/>
3   <nd ref="4241460564"/>
4   <nd ref="1372342252"/>
5   <tag k="cycleway" v="separate"/>
6   <tag k="highway" v="secondary"/>
7   <tag k="lanes" v="2"/>
8   <tag k="lit" v="yes"/>
9   <tag k="maxspeed" v="60"/>
10  <tag k="name" v="Friedenstrae"/>
11  <tag k="oneway" v="yes"/>
12  <tag k="ref" v="Rs 4"/>
13  <tag k="surface" v="asphalt"/>
14 </way>

```

Listing 5.3 Exemplary structure of a way. A way composed of three nodes (via `<nd>` references) with additional tags that further refine the properties of that way. The illustrated way is a highway with a separate cycle way and has two lanes in a single direction. Also, the tags describe the ways name and speed limit.

```

1 <relation id="105785" visible="true" version="9" changeset="55696518"
  ↳ timestamp="2018-01-23T21:16:10Z" user="****" uid="****">
2   <member type="way" ref="27859983" role="from"/>
3   <member type="way" ref="33040473" role="to"/>
4   <member type="node" ref="759219903" role="via"/>
5   <tag k="except" v="psv"/>
6   <tag k="restriction" v="only_straight_on"/>
7   <tag k="type" v="restriction"/>
8 </relation>

```

Listing 5.4 Exemplary structure of a relation. A relation restricts the possible turning directions at an intersection where one way transits into another.



Figure 5.9

Exemplary map excerpt showing that a network is uniquely defined by nodes and edges. Nodes are represented as red dots, while ways are coded as blue lines.

In summary, OSM provides a graph-based map structure that can be formally defined as a directed graph $G = (V, E)$ with V being the set of all nodes and E covering all legs that are found in a map. G is a non-planar directed graph with possible self-loops and parallel edges. Therefore, an edge (i.e. a leg) is a pair of two vertices v_u, v_v that represents a one-way connection between both vertices. However, the set E has to model all connections present in the street network, eventually requiring that bidirectional (two-way) ways be modeled as two reciprocal directed edges (with identical but reversed geometries). v can be part of multiple edges, which promotes it to be an intersection once it is included in at least three edges. An exemplary excerpt of a map is shown in Figure 5.9 illustrating nodes and edges as links between nodes. w.r.t. the figure, one can see that two distinct node pairs define a leg and multiple legs represent a street as it is known in general.

Network
graph

Quality Discussion

5.5.2

Accurate modeling of the real-world street network is of particular importance when working with geospatial information as it is done in Chapters 8 and 12. Inaccurate representations of node coordinates can interfere with map-matching approaches, either based on GPS or IMU data. Therefore, in the following, we will draw attention to some problems that have arisen in the course of this work

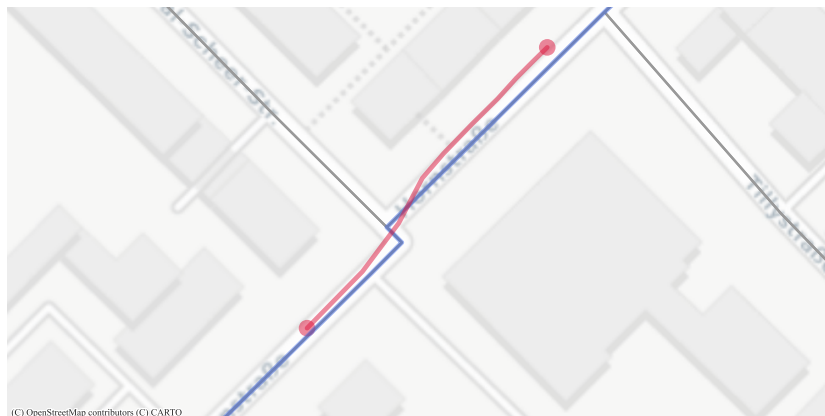


Figure 5.10 **Example of incorrect modeling of the road width.** Inaccurate modeling of the street network results in a gap between the recorded data and the street network.

when working with OSM data. Each problem may require special, anticipatory treatment, which will be discussed in the appropriate sections.

Simplified
representation

The OSM street network is a *model* of the real-world street network. As such, it is a simplified representation of reality; however, the level of abstraction may vary between different models [54]. However, the existing deviation from the real road network is of particular interest in the context of this work, as map matching the recorded trajectories may fail in certain cases if the street network is inaccurate. Figure 5.10 gives an example of such a problem. The road course is shown in the OSM map data (blue) along with a recorded trajectory that passes this road segment (red). It is trivial to grasp that there is a significant difference between the actual trip and the model. This deviation may be due to the lack of a road width model and, consequently, the unrealistic assumption of 90° turns. The excerpt shows two immediate turns, both modeled with 90° . Due to natural driving behavior and the minor change in the road trajectory, the model and the recorded trajectory pose a gap.

Difference
between real
and modeled
world

Another example of such cases is commonly found in lane merges, as shown in Figure 5.11. A lane merge occurs if a road has dedicated lanes for each direction and eventually joins at some point. The ongoing way is then modeled as a single (bidirectional) road. Although G will still contain two e for each direction, they are congruent. Again, the missing road width is challenging, especially in the example presented, where a turn occurs at the exact point of the intersection



Example of simplified modeling of intersections. Inaccurate modeling of the street network results in a gap between the recorded data and the street network.

Figure 5.11

(marked with a cross). There is probably a dedicated turn lane that is missing in the simplified graph. We want to emphasize that such cases imply a significant gap between the actual and the modeled magnetic bearings. The actual bearing delta will be roughly 90° , while the modeled is a sequence of around -30° and 120° .

In addition to modeling, however, driving behavior also has an influence on the quality of the map matching. In this case, again, cornering is problematic, which in reality is usually smoother than represented by the rigid curve radii of the model. The illustration in Figure 5.12 also highlights the problem of inaccurate GPS measurements resulting in a shift between the model's road coordinates and the recorded ones.

Impact of
driving
behavior

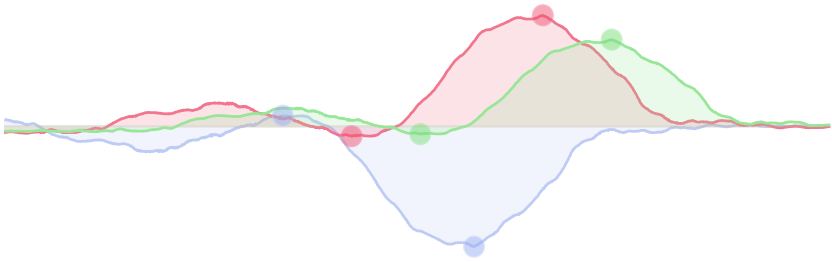
In conclusion, a strict matching process is infeasible in the context of this work. A fuzzy approach is favorable depending on the use case to increase the quality and reliability of the produced data. We will highlight appropriate methods in Chapters 8 and 12.



Figure 5.12 Example of the deviation between model and real-world data. Inaccuracies may occur in both, the recorded GPS data and the map material provided by OSM.

Part II

Privacy by Design-enabled Use Cases



Conventional information systems are composed of multiple integrated and complex processes, each offering essential functionality. At specific points in a process, activities rely on a user identity, which is the representation of a user when he interacts with the system [315].

A user identity may be required and used for different purposes, including, but not limited to, confidentiality, integrity, accountability, and non-repudiation. An exemplary process to access a restricted resource might require a user to identify himself with a service provider (built into the system or offered by a third party) who in turn authenticates a user by verifying the claimed identity (e.g. via a password) and auditing such requests. The procedure is shown in Figure 6.1. The authentication process may be ongoing and reoccurring, i.e. authentication may be needed for additional resources or processes. Consequently, the well-known security goals are used to ensure the correct operation of the system, not only limited to security and safety.

Identification and authentication

During the authentication process, the service provider assigns privileges to a user based on his identity to ultimately authorize the user to perform specific actions. Hence, both steps (identification and authentication) seem inseparable in standard systems. If a user refrains from presenting his identity, he might be excluded from a system, denied certain functionality, or limited to specific processes. Altogether, conventional information systems are challenging to be aligned with requirements for privacy [407] as they historically focus on system integrity rather than client protection [66, 315].

Identity and privileges

However, privileges may be given either on individual or group properties. Individual privileges are commonly attached to properties that a user has. Therefore, he must somehow prove this condition. In contrast, group properties are attached to a group based on properties shared with the group (e.g. age above 18). Thus, a user who claims the respective privileges must show that he is part of a group (e.g. the possession of an identity card in certain countries). Depending on the operation to be performed in an information system, it might be sufficient to use a group-level granularity for an identity. However, the chosen approach directly impacts the viability of the security goals, especially accountability and auditing.

Privileges management and assignment

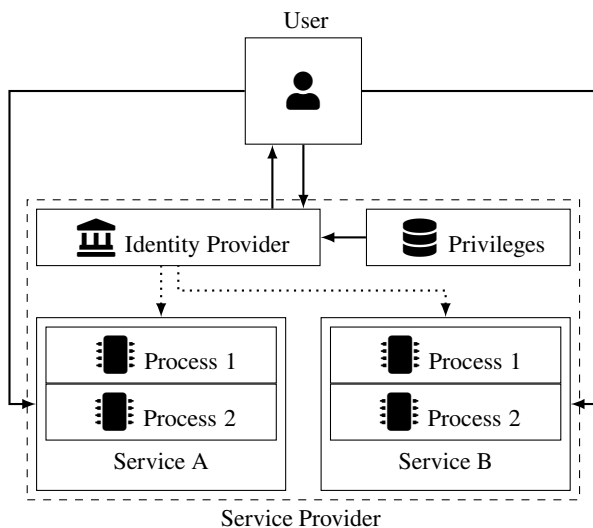
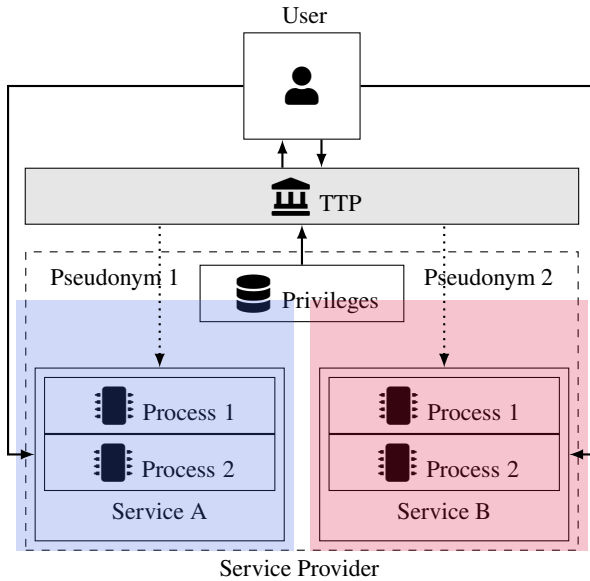


Figure 6.1 Basic architecture of an information system. The information system requires that user identifies himself to the system before he can utilize the services. A user presents his identity to an identity provider who assigns privileges to a user required for service access. (based on Ronald and Borking [315])

A well-known approach to establishing privacy is to rely on a Trusted Third Party (TTP) that serves as a trusted intermediary between a service (which offers the application and the processes) and the user itself. The user may identify himself towards the TTP and receive a pseudo-identity generated by the Trusted Third Party that he may present to the service, as depicted in Figure 6.2, to obtain permissions based on the given pseudo-identity. It is evident that each service receives a separate identity, eventually unable to match them to the same user. Thus, the TTP may offer to convert identities to pseudo-identities but also reveal the linkage between both in case of some misuse. Consequently, the TTP serves both interests by limiting the dissemination of real identities to various information systems while protecting the interests of a service provider, such as accountability or traceability. However, a Trusted Third Party still requires external trust from a user offered, probably through a black-box-based system.

Part II hence focuses on challenges related to complex information systems that need to balance users' privacy interests due to the processing of severely personal data with the aspiration of the quality of service provided by a service provider.



Extended architecture of an information system with a Trusted Third Party. The TTP serves as an identity broker between a user and a service and provides pseudo-identities (pseudonyms) that hold the required privileges to access a service. (based on Ronald and Borking [315])

Figure 6.2

We first focus on the evolution of privacy-aware and privacy-friendly systems in Section 6.1. In fact, we introduce the concept of Privacy by Design that we compare to Privacy Enhancing Technologies (see also Chapter 13). In addition, we will present selected works in Section 6.2. Each work balances the interests of multiple stakeholders, including, but not limited to, privacy and integrity.

Introduction

6.1

We previously outlined that information systems aim to preserve specific security goals that may be controversial depending on the stakeholder interest. Hence, a design decision process must be carried out to find a trade-off between conflicting requirements. For example, tracking of all user interactions in an e-Commerce shop allows the individual recommendation of products depending on the interests of a user considered for the possible realization of cross-selling. It

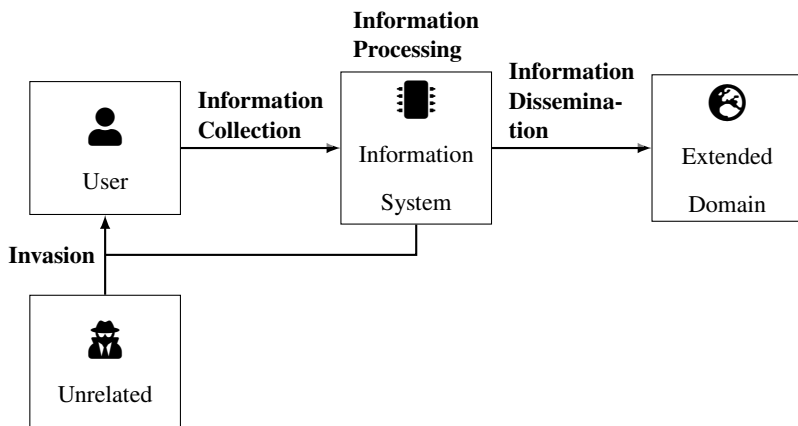


Figure 6.3 Relationship of different groups of harmful activities to the privacy of a user. Threats may arise at each of the four activities requiring specific countermeasures to be taken into consideration. (based on Solove [349])

is reasonable to assume that this behavior is in the interest of the service provider. However, a user might disagree with being thoroughly tracked during the online shop experience.

6.1.1 PETs

Steps of
information
disclosure

A user's primary interest may be self-management of data and a reduction of his digital footprint to preserve his privacy. However, the previous example illustrates that privacy problems may occur in different information system processes due to exhaustive information disclosure, either direct or indirect. Processes can be categorized according to the taxonomy of Solove [349], that is, information collection, information processing, information dissemination, and invasion. Information collection is the process of collecting data from users, either directly or in a hidden way, while information processing is the downstream analysis, evaluation, and manipulation of data. Information dissemination describes how information may leave a process found in the information system or the information itself—not necessarily unintentional. Invasions “*are direct interference with the individual*” [350], for example, limiting the number of products visible in the said online shop. The interrelationship and interaction of the various steps is illustrated in Figure 6.3.

This circumstance describes the imperative why privacy should also be an important goal in the design of information systems. Privacy can only be established by allowing users to shape and manage their identities modeled in a system. Crucial, therefore, are the features that allow users to steer aspects according to Solove [351], such as *Limit on Power*, *Reputation Management*, *Maintaining Appropriate Social Boundaries*, *Trust*, *Control Over One's Life*, *Freedom of Thought and Speech*, *Freedom of Social Political Activities*, *Ability to Change and Have Second Chances*, *Protection of Intimacy*, *Bodies*, and *Sexuality* and *Not Having to Explain or Justify Oneself* (c.f. Section 1.2.1).

Self-realized
privacy

However, this management functionality is complex and challenging to provide and may interfere with the business intention of a service provider. External techniques are a way out of this dilemma. With the help of Privacy Enhancing Technologies, a user can interact with a compatible system with his demands being enforced without the cooperation of a service [129]. PETs allow one to manage one's identity, or more precisely, the identity perceived by a service. For example, data minimization can be enforced through PETs as it may review submitted information for being detrimental to privacy or control and suppress information disclosure at all [377]. It is trivial to grasp that PETs interfere significantly with the functionality of a service, eventually deteriorating the user experience to the extent that no meaningful use is possible. Moreover, they are often targeted for specific information systems. We focus on PETs in Chapter 13.

Third-party
assistance

Privacy Aware System Design

6.1.2

An evolutionary construct is called Privacy by Design which fundamentally differs from PETs as the latter works *ex-post* without being service agnostic. PETs are *privacy design patterns* that solve problems with a non-privacy-aware system design [160]. In contrast, Privacy by Design is a *privacy design strategy* [160, 174] defining that systems should be designed to provide all functionalities while ensuring privacy and personal control over one's information [66]. Therefore, an information system designed with privacy as a first-level requirement is more inclusive, as it provides the full range of functionality while ensuring privacy and providing personal control over one's information [66]. Privacy by Design was invented by Cavoukian [66] composing general and holistic guidelines to design the interaction of information systems, accountable business practices, and infrastructure. It is also subject of the General Data Protection Regulation.

The seven principles of a privacy-aware design approach are listed in the following.

Seven
principles for
privacy by
design

1. A Privacy by Design-based architecture is designed with proactive protective measures in mind so that the probability for privacy-invasive events is negligible, eventually driving remedies for data breaches obsolete.
2. Privacy has to be enabled for any user *by default* so that all users, independent of background, setting, or interaction with the system, are architecturally protected.
3. The architecture should consider privacy as a core functionality deeply integrated into the information system without diminishing any experienced services.
4. The experience itself is of particular interest for a user; hence, a Privacy by Design-based approach should provide full functionality while preserving privacy unrestrictedly.
5. As an information system is composed of multiple processes and services, Privacy by Design demands that the whole lifecycle is extensively considered and thoroughly protected.
6. Such a system should demonstrably and comprehensibly prove interoperability independent of a stakeholder; this includes, without limitation, the full protection of data.
7. User-centricity is essential so that the interests of a user providing the data are equally considered with other design decisions i.a. driven by the service provider

These principles may be considered abstract, high-level, and open to interpretation [377]. Their organizational and technical realization depends on the specific information system and varies with the sensitivity of the data [66, 160]. However, it is still worth mentioning that Privacy by Design requires the design process to take into account all the privacy and derived requirements to ensure the named principles. In fact, privacy should not be considered to be added to an existing architecture [315]. We will present our interpretation in Section 6.2 and the rest of Part II.

6.1.3 Data Minimization

The privacy taxonomy presented by Solove [349] illustrates four groups that specifically threaten user privacy. Indisputable, minimizing the amount of (personal) data transmitted to a service provider also reduces the impact of a privacy violation. Therefore, *data minimization* is a desirable path in the design of infor-

mation systems. There are misunderstandings regarding the definition of data minimization, though, as Gürses et al. [160] points out. In the classical sense, it can be understood that only necessary data should be transmitted, eventually resulting in sensitive data made available throughout the information system (within hostile domains). Even though sensitive data will be available at some point in the information system, it is favorable to be kept within the control of a user [297] probably in encrypted or aggregated form [160]. We stick to the second definition and argue that raw data should never leave the domain of a user, as it enables multiple side-channel attacks, some of which are focused in Part III of this work.

As a consequence, the use of encryption or similar methods before transmitting data to a service provider (we will leave the question of the transmission of the identity aside at this point) is not intended to achieve Privacy by Design. Still, this does not necessarily yield any minimization of data. In fact, data minimization strategies are favorable when designing a Privacy by Design and by Default system. However, a privacy-friendly system must not abandon the basic security features of an information system that were motivated at the beginning of this chapter. Gürses et al. [160] defines data minimization strategies as follows:

Data
minimization
strategies

- ▶ **Minimize Collection:** whenever possible limit the capture and storage of data in the system
- ▶ **Minimize Disclosure:** whenever possible constrain the flow of information to parties other than the entity to whom the data relates
- ▶ **Minimize Replication:** whenever possible limit the amount of entities where data is stored or processed
- ▶ **Minimize Centralization:** whenever possible avoid single point of failure in the system
- ▶ **Minimize Linkability:** whenever possible limit the inferences that can be made by linking data
- ▶ **Minimize Retention:** whenever possible minimize the retention of data in the system

Consequently, it is a matter of combining different strategies accordingly to achieve Privacy by Design. In Section 6.2 we apply these strategies to specific data-intensive problems with high abusive potential. Most of the strategies may also be (implicitly) found in General Data Protection Regulation (GDPR) Article 5 as data minimization is an essential concept within this regulation.

Significance
for research

6.1.4 Designing Privacy by Design

The design of an information system is composed of multiple steps, each of which requires complex decisions concerning user privacy and system integrity. Due to the holistic approach of Privacy by Design, the strategies can already be considered initially and are driving elements of the development. To successfully apply the strategies, a four-step interlocked stage model is proposed [160] that is discussed in the following.

Separate the domains

First, the relevant entities must be defined and assigned to a domain. Domains are either user-controlled or service-controlled, with the latter including all data processors and data controllers (refer to GDPR for a formal definition of both roles; see also Section 9.2). The information system with multiple processes can span both domains. Furthermore, the service provider (or the authorizing instance) is typically found in the service domain.

Identify required data

Most information systems create value based on data provided by a user. Therefore, the second step is to define the data necessary to provide a service and identify the scope of data required to be available in a service domain, eventually leaving the user domain. This task is explicitly challenging, although even the GDPR restricts the amount of data to be collected to the data required to fulfill a service. However, the boundaries are floating at this point, mainly due to sophisticated ML applications that benefit from extensive data sets. Thus, service providers tend to a “collect-all-data” approach or “select-before-collect”, although these procedures require a rethinking. It is no longer applicable when sticking to the Privacy by Design principles that demand purpose-driven data collection.

Map domains to data based on privacy invasiveness

The identified scope and amounts of data required to provide a service are considered in the next step to define the distribution of data within an architecture. We have learned that information systems are composed of multiple processes, each providing functionality that, in combination, define a service. The initial step identified the two domains, namely the service domain and the user domain, while the second step highlights the data needed at specific points in the information steps. At this point, a mapping has to be done that aligns the two insights but does not forget the credo of data minimization. Only necessary data should leave the domain of a user. This can also be a two-step approach where data flows through both domains, although sensitive data may be processed in the user domain and is then forwarded in a “not-so-privacy-invasive style” to the service domain.

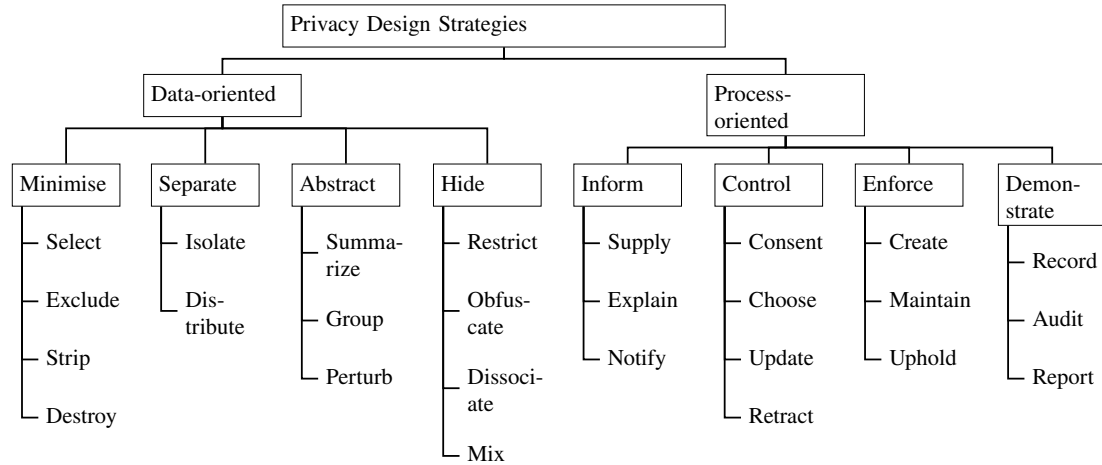


Figure 6.4 Privacy design strategies are composed of methods that target the data itself and the process. There are multiple methods to select from for each group. (based on Hoepman [174])

Implement
privacy-
friendly
strategies

Finally, an architectural draft is composed that critically questions the quantity of data and processing steps, although it does not include any of the privacy-enforcement strategies explained previously. However, the architecture of the information system can now be enriched with privacy design strategies intended to reduce further the privacy threats listed by Solove [349]. Privacy design strategies can be a comprehensive list of privacy design strategies is given by Hoepman [174], which are separated into data-oriented and process-oriented strategies and match the presented strategies. Data-oriented strategies address privacy-friendly processing of the data itself and include methods to minimize data, separate personal and consumption data¹, abstract the level of detail or hide specific aspects. Process-oriented strategies include organizational aspects of an information system, such as the need to inform a user, provide him adequate control, allow him to enforce his privacy interest, or demonstrate privacy-friendly data processing. Figure 6.4 lists the appropriate methods for each group.

Contradict-
ing example

An example that does not follow the Privacy by Design principle is Pay-How-You-Drive. It uses sensor data collected in the user domain to assess the driving behavior of a person based on raw data that is sent to a centralized data processor and eventually replicated to other entities. All data is inextricably linked to the user who collected the data as he is billed accordingly. Raw data allows for a variety of privacy violations. This topic is extensively discussed in Parts III and IV of this work.

6.1.5 Inclusive Design

Apart from a privacy-friendly architecture, the presentation toward a user is also of particular interest. We claim that a Privacy by Design-designed information system should be open and inclusive to be in line with the design principles defined by Cavoukian [66].

Dark
patterns

The information requester might also substantially impact the sender's disclosure behavior. It is essential to remember that information systems address users with different backgrounds. By appropriately compounding the risks associated with the information disclosure process of sharing data from the user domain, users

¹Consumption data includes all the information that is required to provide application functionality. For instance, route planning software requires start and destination to calculate a route. In contrast, personal data, such as the credit card number, may be used to accordingly bill for the service.

are encouraged to dismiss any concerns automatically. Risks are minimized by design so that companies can realize a “collect-all-data” mentality [167, 351]. In this context, we speak of dark patterns, in which the user’s decision-making process to weigh the disclosure of data is influenced by the design of the information system itself [29, 196]. The decision-making process per se is already a violation of the Privacy by Design paradigms in that privacy is no longer omnipresent by default in this case. In addition, non-transparent communication deliberately tries to deceive the user. Friction aims at something similar, a process in which the convenience of a user to protect his data is made as burdensome as possible [260]. Moreover, privacy settings are often minimal, while preferences for data dissemination are encouraging.

Such behavior conflicts with the Privacy by Design paradigm in multiple cases and must be avoided. The information disclosure process is severely complex as it is influenced by the benefits and costs, the context in which it takes place, and the cognitive perception of an individual [407]. Consequently, an information system based on Privacy by Design should be open and inclusive. In fact, it should make any decision process concerning information disclosure obsolete once a user decides to interact with an information system in the first place.

Avoid
decision
making

Privacy-friendly Research

6.2

When composing an information system, the Privacy by Design paradigm is an essential construct to consider. It is based on the privacy concerns that we introduced in Section 1.2. However, the path followed is different since, as described, systems constructed according to the paradigm do not require the user to make any additional decisions other than the fundamental question of participation. Thus, the systems presented may take into account the user requirements accordingly and be designed according to best practice [407].

Overview of Research

6.2.1

Multiple research was conducted in the field of Privacy by Design that explicitly focuses on privacy-friendly and privacy-aware methods and approaches for processing user-related information. This section briefly lists the relevant work, also shown in Figure 6.5. In total, seven papers have been published as peer-reviewed work, two of them being extended versions.

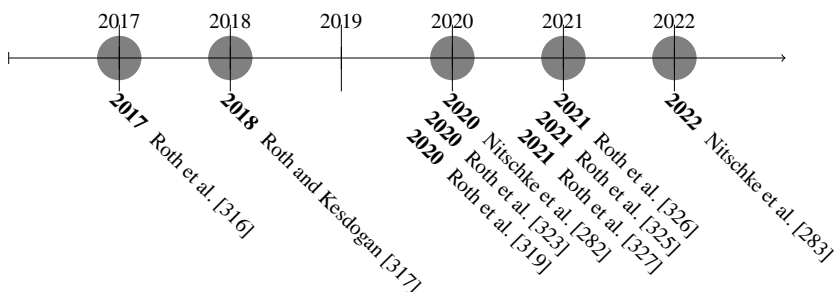


Figure 6.5 **Research published in the field of Privacy by Design.** In total, nine publications either present the Privacy by Design paradigm or implement it in different settings.

PARTS - Privacy-Aware Routing with Transportation Subgraphs [316]

Authors: Christian Roth, Lukas Hartmann, and Doğan Kesdoğan

Conference: Nordic Conference on Secure IT Systems (NordSec)

Publication Date: November 2017

Abstract: To ensure privacy for route planning applications and other location based services (LBS), the service provider must be prevented from tracking a user's path during navigation on the application level. However, the navigation functionality must be preserved. We introduce the algorithm PARTS to split route requests into route parts which will be submitted to an LBS in an unlinkable way. Equipped with the usage of dummy requests and time shifting, our approach can achieve better privacy. We will show that our algorithm protects privacy in the presence of a realistic adversary model while maintaining the service quality.

A Privacy Enhanced Crowdsourcing Architecture for Road Information Mining Using Smartphones [317]

Authors: Christian Roth and Doğan Kesdoğan

Conference: International Conference on Service-Oriented Computing and Applications (SOCA)

Publication Date: November 2018

Abstract: The digitization of our road and traffic systems enables evermore advanced location based services to support us in our everyday tasks with prominent examples being navigation applications like Google Maps or speed camera

directories like TomTom Speed Cameras. The information collection that lies at the base of these applications however is often either done behind closed doors, or relies on the goodwill and time investment of voluntary community members providing such information as best they can. In this paper we present a new crowdsourcing architecture for voluntary road and traffic system data collection, that on the one hand values and protects the privacy of the participating community members and on the other hand significantly eases their manual workload by detecting and inferring applicable information through the sensors of their mobile phones using a self-created Android application. Our approach shows reliable results for the road system properties we defined. We propose an enhancement for the route navigation process by including the acquired road information.

Harmonized Group Mix for ITS [282]

Authors: Mirja Nitschke, Christian Roth, Christian Hoyer, and Doğan Kesdoğan

Conference: International Conference on Information Systems Security and Privacy (ICISSP)

Publication Date: February 2020

Abstract: Vehicle-to-Vehicle (V2V) communication is crucial for almost all future applications in the context of smart traffic, such as autonomous driving. However, while current standards like WAVE provide a technical platform for communication and management, they lack aspects of privacy for their participants. In this paper, we introduce a Harmonized Group Mix (HGM), an architecture suited to exchange information in ITS, compatible with current standards. HGM does not rely on expensive Road-Side-Units (RSUs) or complex organizational relationships to introduce a trust anchor but is built on the concept of peer-to-peer networks. Hence, our proposal does not require any changes to current environments and is eventually easy to deploy in the real world. Our proposed method provides k -anonymity using group signatures and splits trust between multiple parties. At the same time, the integrity of the system is preserved. We evaluate our approach using the simulation framework Veins. Our experiments show that HGM is feasible from a performance and privacy perspective in the given context.

iTLM: A Privacy Friendly Crowdsourcing Architecture for Intelligent Traffic Light Management [323]

Authors: Christian Roth, Mirja Nitschke, Matthias Hörmann, and Doğan Kesdoğan

Conference: International Conference on Data Science, Technology and Applications (DATA)

Publication Date: July 2020

Abstract: Vehicle-to-everything (V2X) interconnects participants in vehicular environments to exchange information. This enables a broad range of new opportunities. We propose a self learning traffic light system which uses crowdsourced information from vehicles in a privacy friendly manner to optimize the overall traffic flow. Our simulation, based on real world data, shows that the information gain vastly decreases waiting time at traffic lights eventually reducing CO2 emissions. A privacy analysis shows that our approach provides a significant level of k-anonymity even in low traffic scenarios.

CrowdAbout: Using Vehicles as Sensors to Improve Map Data for ITS [319]

Authors: Christian Roth, Ngoc Thanh-Dinh and Doğan Kesdoğan

Conference: International Conference on Social Networks Analysis, Management and Security (SNAMS)

Publication Date: December 2020

Abstract: Crowdsourcing can be seen as an opportunity to provide important information for Intelligent Transportation Systems to improve the service quality of various applications in this domain. Autonomous or assisted vehicles need the most accurate map data possible to adjust the respective assistants to it. In this work, we present CrowdAbout, a system that uses the crowd as mobile sensors to collect data from smartphone sensors during trips. The system recognizes special traffic events like roundabouts with the help of machine learning. These findings are used to automatically correct OpenStreetMap data and adapt them to a changing road network. An evaluation of different machine learning algorithms using self-collected realworld data of over 200 roundabouts shows that the recognition of roundabouts including exit and radius is possible with high accuracy.

ROADR: towards road network assessment using everyone-as-a-sensor [326]

Authors: Christian Roth, Ngoc Thanh-Dinh, Markus Hornsteiner, Verena Schröppel, Marc Rossberger, and Doğan Kesdoğan

Conference: International Conference on Distributed Sensing and Intelligent Systems (ICDSIS)

Publication Date: July 2021

Abstract: Complete and up-to-date map data plays a critical role in many contemporary and future applications such as autonomous driving level 3+. In terms of crowdsourcing, a data basis can be created that meets these stringent requirements without dedicating additional resources. With ROADR, we present a holistic platform to gather knowledge about a road network and its properties to further enhance either semantic or syntactic information. The privacy-by-design platform uses a smartphone application to collect crowdsourced data and performs local machine learning. Only less sensitive data is forwarded to a centralized platform that aggregates and processes information from the crowd to provide value-added information found in a vehicle's trajectory. Also, the paper provides a thorough analysis of the respective Floating Phone Data indicating two exemplary events, namely traffic light and traffic circles. Our evaluation shows that the recognition is done in real-time but in a resource-efficient way.

iTLM-Q: A Constraint-Based Q-Learning Approach for Intelligent Traffic Light Management [325]

Authors: Christian Roth, Lukas Stöger, Mirja Nitschke, Matthias Hörmann, and Doğan Kesdoğan

Journal: Data Management Technologies and Applications (Communications in Computer and Information Science)

Publication Date: July 2021

Abstract: Vehicle-to-everything (V2X) interconnects participants in vehicular environments to exchange information. This enables a broad range of new opportunities. For instance, crowdsourced information from vehicles can be used as input for self-learning systems. In this paper, we propose iTLM-Q based on our previous work iTLM to optimize traffic light management in a privacy-friendly manner. We aim to reduce the overall waiting time and contribute to a smoother traffic flow and travel experience. iTLM-Q uses Q-learning and is constraint-based in such a way that no manual traffic light cycles need to be defined in advance, hence, being able to always find an optimal solution. Our simulation-based on real-world data shows that it can quickly adapt to changing traffic situations and vastly decrease waiting time at traffic lights eventually reducing CO2 emissions. A privacy analysis shows that our approach provides a significant level of k-anonymity even in low traffic scenarios.

STRIDE: Secure Traffic Reporting Infrastructure based on Distributed Entities [327]

Authors: Christian Roth, Marc Rossberger, Christoph Schreyer, and Doğan Kesdoğan

Conference: International Workshop on Smart Cities Systems Engineering (SCE)

Publication Date: December 2021

Abstract: Efficient and intelligent traffic networks rely on the constant exchange of information between participants. For instance, navigation services benefit directly from the availability of real-time traffic information to suggest the most time-optimized and ecologically sustainable routes. This type of information is now commonplace and is formed based on extensive, microscopic movement profiles. This imposes direct constraints on the location privacy of users who implicitly or explicitly share such information. In this paper, we present STRIDE as a component of an ITS to gather real-time traffic information in a privacy-friendly manner, ultimately protecting data sources (i.e., users) against data misuse. Our architecture is designed around the concept of distributed trust, preventing attackers from tracking vehicles across the network, even if they succeed in compromising network components. We also achieve conformity to ETSI standards and conclude that real-world implementation of our architecture would be feasible. Thus, we evaluate STRIDE using SUMO and a real-world data set to analyze STRIDE's potential to provide accurate traffic information. Furthermore, we show that STRIDE ensures k-anonymity even in sparse traffic scenarios, eventually protecting location privacy of each vehicle.

Harmonic Group Mix: A Framework for Anonymous and Authenticated Broadcast Messages in Vehicle-to-Vehicle Environments [283]

Authors: Mirja Nitschke, Christian Roth, Christian Hoyer, and Doğan Kesdoğan

Journal: Information Systems Security and Privacy (Communications in Computer and Information Science)

Publication Date: January 2022

Abstract: Nowadays Vehicle-to-Vehicle communication (V2V) plays an increasingly important role, not only in terms of safety, but also in other areas of Intelligent Transport Systems (ITS). However, privacy is often underestimated in this context. In this paper we describe an extended version of our Harmonized Group Mix (HGM). HGM has the objective of enabling the privacy-friendly data exchange between vehicles in an ITS without neglecting other requirements

such as integrity. In contrast to other approaches a complex organizational structure is not required and HGM is thus easily applicable. Rather, the idea of a Mix system is transferred to ITS communication, but the ITS-specific real-time requirements can still be met. The simultaneous use of group signatures can ensure a high degree of k-anonymity and prevent the tracking of participants. A distributed knowledge approach provides trust but at the same times allows revealing fraudsters. In addition to a detailed security analysis, this paper evaluates the approach using the simulation framework Veins and focuses on the exact vehicle movements and the groups formation respectively changes over time and their influence on each other.

Aspects of Research

6.2.2

We now present the concepts behind the research done concerning Privacy by Design. The topics are related to the processing of sensitive sensor data, but on the other hand, they also place a strong emphasis on mobility as it occurs in transport networks with different participants. This often results in a dichotomy of various security goals depending on the respective stakeholder. Commonly, privacy conflicts with integrity (or accountability and auditability), but as we will show, such antagonisms may be solved using a distributed system design.

Building on the findings of this chapter, the Privacy by Design concept is achieved through a reasonable division of data processing in the corresponding domains. Questioning the information flows and the data required to implement the services is another crucial building block. The principle of data minimization is ensured by incorporating data minimization strategies in accordance with Gürses et al. [160]. An analysis of our work is presented in Table 6.1. The table introduces the included stakeholders and the data that is processed through the information system. A location is known from this work's notation (i.e. *loc*) while a trajectory is an ordered sequence of locations. The sensor data itself is defined in Section 5.2. Moreover, an assessment of the ability to achieve the data minimization strategies is given in the table. Eventually, various privacy design strategies [174] are essential building blocks to achieve Privacy by Design and Privacy by Default in combination with high service quality. The corresponding details are presented in the following chapters Chapters 7 and 8 which present two information systems in more detail.

Structural
approach

Table 6.1 Mapping of publications in the field of Privacy by Design and their applied privacy design strategies. Depending the processed data and stakeholders, different minimization strategies can be fully achieved (○), achieved to some extend (◐), or not ensured at all (●).

Ref	Stakeholder	Data	Co ¹	Di ²	Re ³	Ce ⁴	Li ⁵	Re ⁶
[316]	User, Service	Location	●	●	○	◐	○	●
[317]	User, Identity Provider, Service	Location, Sensor Data	●	○	○	◐	○	●
[319]	User, Service	Location, Sensor Data	●	○	○	○	○	●
[326]	User, Identity Provider, Service, External	Location, Sensor Data	●	○	○	○	○	○
[325]	User, Identity Provider, Service	Trajectory	●	○	○	○	○	●
[327]	User, Identity Provider, Service	Trajectory	●	○	○	○	○	●
[283]	User, Identity Provider, Service	Generic Data	●	◐	○	○	◐	●

¹ Collection ² Disclosure ³ Replication ⁴ Centralization ⁵ Linkability ⁶ Retention

Common
research
problem
structure

Research shares a common structure across different problem domains: Users provide, generate, or possess data that an information system requires to provide added value. In general, the more accurate the user data, the better the service quality may be. The information system may also require authentication or authorization depending on the given task; hence, identification is needed in the first place. User-acquired information may be processed, aggregated, or analyzed and persisted in a database or forwarded to external stakeholders. In summary, relying on user data that originates from a user's private and sensitive domain poses a threat to privacy at multiple steps [349]. Hence, a user's privacy has to be protected while ensuring the integrity and reliability of the information system, eventually being a dichotomy to be balanced.

Overview of
research

As illustrated in Table 6.1, data collection is not minimized in almost all works, as data is collected within the user domain without restriction. Local computations are performed in the user domain, which is expected to be trustworthy. However,

disclosure, replication, and centralization are minimized. Being able to link multiple interactions with single or multiple information systems enables an adversary such as a curious information system provider to generate detailed user profiles. Thus, this ability has to be mitigated. Retention is hard to achieve as the submitted data of a user is disseminated to the information system, eventually leaving his control. Admittedly, a thoughtful disclosure process may prevent any threats related to retention at all.

It is vital to understand the difference between collection and disclosure. The former does not compromise privacy, even if it is done to a large extent. The data that can be collected in the user domain is generated there continuously. According to Solove [350], as long as the data does not drain out of the trusted domain, there is no harm because the information processing is local, and there is no information dissemination at all. Therefore, a core concept of the Privacy by Design research lies in the strict separation of domains and visibility that is dominantly present in the presented works.

Difference between collection and disclosure

Multiple approaches are meaningful depending on the intention.

Collection The *collection* process is extensive but can be partially limited by using other sources of data that are already used to draw similar inferences [319, 326]. Limiting the usage of resources also limits the probability of a breach.

Building blocks found in the research

Dissemination The *dissemination* of data can be minimized by not sending the data and performing data-intensive applications in the user domain and only sending aggregated results from these computations [319, 326]. This is a feasible approach to processing sensor data with high side-channel attack potential (c.f. Chapter 10). In addition, the limitation of leaked data can also be achieved by applying privacy-protective cryptographic protocols [160] such as zero-knowledge proofs [327] or attribute-based credential systems [325].

Replication Furthermore, the *replication* of the data can be limited by applying encryption on the data so that only a specific party can evaluate the data [283, 317, 325, 327]. According to the chosen encryption scheme, the service domain cannot access the data unless a key is given (perhaps in terms of proving innocence by a user) [283] or perform some calculations using e.g. homomorphic encryption [317, 327].

Centralization Relying on a *centralized* infrastructure may allow a single entity to map activities on users, ultimately violating privacy requirements. Sophisticated credential systems (c.f. Sidebar D) distribute identity verification, authorization process, and fraud investigation between multiple and distinct entities [325–327]. Such systems may also provide mechanisms to increase the

traceability and transparency of operations between different entities. Increases in centralization thus become visible to users. Centralization can only be limited if a dedicated and unique service is required [316] which further stresses the need to keep an eye on the other principles of data minimization.

Linkability *Linkability* can be limited by refraining from presenting a static identity throughout the interaction with an information system. As originally mentioned, permissions can also be granted to a user based on shared properties in a group. Therefore, showing the affiliation to a group allows one to hide an identity using group signatures [74] which is a meaningful approach [283]. Additionally, the threat of linkability can be reduced if an information system cannot map the activities of a user to incoming requests. Mix networks [75] may be used for this purpose [283].

Retention Last, the *retention* of the disseminated data is out of the control of a user apart, but an information system's design can ensure reduced persistence duration. For example, submitted information can be aggregated on the remote side with the summants being disposed afterwards [326].

Outlook In the rest of this part, we will present two selected works. Both aim to balance the privacy and integrity of data closely associated with the user from whom they originate. Moreover, the sensitivity and granularity of the data could enable severe abusive potential, such as creating exhaustive movement profiles.

The use of the road network is constantly increasing, which is affecting the efficiency of the infrastructure and related applications. As a result, reliable traffic management is critical for congestion control. At the moment, navigation systems attempt to plan the ideal, personal, and microscopical route depending on the underlying street network structure and utilization. As a result, they rely on current information. However, such solutions are rarely sustainable, as the ideal path is determined based on variables that are the same for all participants. Thus, navigation instructions may contribute to the deterioration of the real efficiency of a route. Using collaborative routing approaches such as NUNAV¹ gracefully distribute user-recommended routes, reducing congestion. However, they rely on even more up-to-date traffic data than traditional systems. Information is analyzed at a central service because it requires a global, macroscopic picture of the traffic condition, allowing the determination of data with a high degree of integrity. In addition, these technologies must be continuously available to enable intelligent routing.

Therefore, permanent surveillance of all road users moving within vehicular networks is desirable and is built into Intelligent Transport System infrastructures by default. However, the privacy of participants suffers significantly in such a system since the global service provider may track the paths of cars and infer additional information from them. Confidentiality is out of the question, despite the fact that it would benefit user acceptability. However, complete security and privacy objectives must be considered for these vehicular networks and particularly the unique use case of real-time traffic reporting. Additionally, the participants' identities should be kept confidential to safeguard user privacy.

Conditional
permanent
monitoring

¹<https://www.nunav.net/>

Acknowledgement

Parts of the research presented in this chapter are based on work published previously [327].

STRIDE As a result, we introduce a novel secure ITS architecture called *STRIDE* (Secure Traffic Reporting Infrastructure based on Distributed Entities) that outputs correct traffic information while maintaining user privacy. It achieves this by building on the standardized European Telecommunications Standards Institute (ETSI) design, which enables implementation on current Vehicle-to-Anything (V2X) networks. We make further adjustments to increase performance and protect user identities from insiders.

Contribution This chapter shows how to balance the conflicting interests of integrity and data quality to provide unrestricted functionality by following the Privacy by Design principles presented in Chapter 6. In fact, we contribute with

- ▶ a thorough system analysis and overview of state-of-the-art ITS architectures,
- ▶ a theoretical concept to derive real-time traffic information in a privacy-friendly manner which is easy to integrate with existing V2X standards,
- ▶ a microscopic simulation based on real-world data using SUMO, and
- ▶ a full evaluation of the system analyzing integrity and privacy requirements, including attacker consideration.

Structure The remainder of this chapter is structured as follows. First, we present related work in Section 7.1. Subsequently, an overview of V2X architectures is given in Section 7.2 including crucial security requirements and privacy aspects. Section 7.3 presents the proposed architecture *STRIDE* to collect information about the utilization of the street network in a privacy-friendly manner. In Section 7.4, we explain how the simulation was conducted, which is the basis for our evaluation in Section 7.5. The evaluation focuses on the performance and privacy aspects of the architecture. Finally, this chapter is concluded in Section 7.6.

7.1 Related Work

Integrity-
ensured ITS
applications

The growing volume of positional data allows various applications in traffic monitoring systems. However, incorporating location-based services into ITSs remain a difficulty for current research: the given services must be available in real-time, resistant to manipulation, and protective of their users' privacy. The usage of pseudonyms maintained by a Public-Key Infrastructure (PKI) [52,

295, 379, 401] has been a widely adopted way of incorporating privacy into the design of ITSs. This system offers safe communication between participants and tries to maintain anonymity by exchanging pseudonyms on a periodic basis to prevent user tracking. However, this method has one disadvantage: the constant exchange of new pseudonyms, certificates, and Certificate Revocation Lists creates a communication overhead in the system, requiring users to consume additional processing power and maintain a constant network connection, which can be difficult for moving participants.

To alleviate user burden during subsequent certificate requests, Brecht et al. [52] and Whyte et al. [401] propose a butterfly key expansion-based technique to create numerous key pairs simultaneously. By segregating certificate ownership knowledge between two authorities, the system preserves user privacy while maintaining responsibility. Verheul et al. [379] uses a similar technique called Issue-First Active-Later (IFAL) to extract numerous secret keys from a single key pair, allowing players to construct several temporary pseudonyms independently. Activation codes are handed out periodically and are necessary for authentication in this technique. By omitting the transmission of activation codes to misbehaving automobiles, the use of Certificate Revocation Lists can be reduced even further.

Privacy-friendly key exchange

While these tactics defend against simple tracking efforts, they are insufficient to safeguard users' privacy against more sophisticated assaults. For example, attackers can utilize supplementary data about individuals in conjunction with statistical analysis to forecast movement patterns and match them to pseudonymous GPS traces [217]. This attack vector is particularly severe within an ITS since cars are required to submit their speed regularly and position data to a processing unit, as Hoh et al. [176] demonstrate.

Lack of privacy features

Multiple designs have been suggested to adequately protect user privacy, including [154, 177, 178]. Most of these systems are based on a central organization assigned with the responsibility of cloaking user data in order to achieve *k-anonymity*. Other systems seek to improve privacy by sending only partial trajectories [76] or by introducing statistical noise to aggregated data to introduce inaccuracies [119].

TTP-based privacy enforcement

By leveraging decentralized geographically veiled traffic reporting and knowledge separation between its components, the proposed *STRIDE* solution ensures privacy by design, as any (compromised) unit cannot monitor users. Additionally, it utilizes IFAL to increase efficiency and strengthen its defenses against

Our proposal

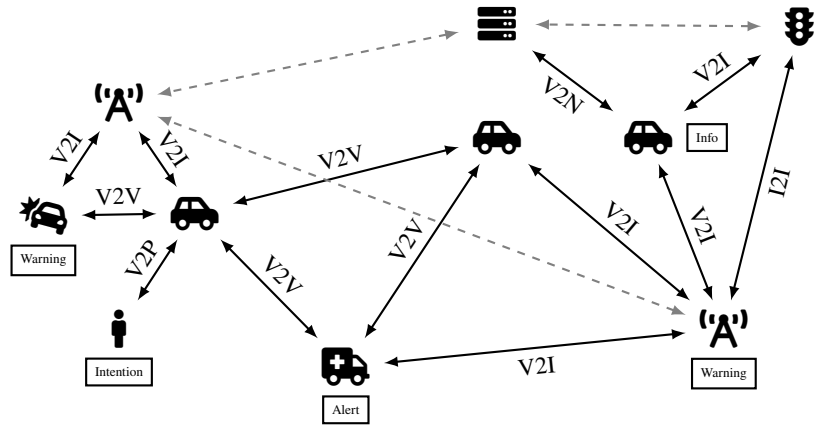


Figure 7.1 A V2X network is composed of heterogeneous entities. It is composed of multiple entities spanning a distributed network that includes various communication channels (denoted as \longleftrightarrow) including out of band approaches (represented by \dashrightarrow).

Sybil attacks. Furthermore, due to its conformity to the ETSI network design guidelines, it is compatible with existing V2X networks.

7.2 Vehicle-to-Anything

Constantly communicating vehicles

The automobile industry's development has led to more intelligent cars containing more integrated sensors and driving assistants, enabling autonomous driving. To further utilize these capabilities and increase traffic efficiency and safety on roads, traffic participants partake in V2X communication. This includes vehicles sending, for example, FCD to other vehicles (Vehicle-to-Vehicle) or the road infrastructure (Vehicle-to-Infrastructure) comprised of Road-Side Units (RSUs). Data includes vehicle's velocity v , location loc , and current time slot t . The vehicle authenticates towards the RSU by also sending its unique identity I or a temporary pseudonym p depending on the employed architecture. As defined in the directive 2010/40/EU of the European Union, this system is called an ITS and can be used for traffic management, resulting in reduced travel times and increased road safety.

ETSI en 302
663 ITS-g5

Figure 7.1 shows the structure of an ITS with participants and communication channels as an example. Depending on the respective partner and the addressed so-called subsystem, the communication channel is named accordingly, with

V2X being used as a generic collective term. Information is relayed between participants in a decentralized, peer-to-peer-based way using V2X communication standards (e.g. ETSI EN 302 663 ITS-G5) denoted by \longleftrightarrow , although out-of-band communication, shown as $\leftarrow\rightarrow$, may also be present for interconnecting services. The following summary is based on Alam et al. [8]

As illustrated in Figure 7.1 different entities may freely join, openly participate and seamlessly communicate in a V2X network. They appear on par in the ITS. An Intelligent Transport System Station (ITS-S) is a fundamental building block that enables the execution and usage of different ITS applications. The four communicating parties (ITS-S) within the ITS are **personal**, **vehicle**, **central**, and **roadside** entities (also called RSU). The entities represent individuals and pursue their interests or undertake collaborative and dedicated tasks. Furthermore, an ITS-S may be connected to local systems. For example, a vehicle ITS subsystem may consist of an ITS-S capable of communicating with an ECU or additional sensors. All entities are also found in Figure 7.1.




ITS
subsystems
and
Intelligent
Transport
System
Station

The ITS-S reference architecture (c.f. ETSI EN 302 665) defines four different functional components that may be present in an ITS subsystems. The underlying concept is a layer-based system, similar to the International Organization for Standardization/Open Systems Interconnection model, which is grouped based on functionality. The architecture has management and security components in addition to the access, network/transport facilities, and applications layers, which are necessary for functionality (c.f. ETSI EN 302 665). First, the **ITS station host** enables a personal device to access ITS applications. Next, a **ITS station gateway** is able to convert protocols and enable accessing external, proprietary networks. The **ITS station router** is needed to connect to other ITS-Ss, so it interconnects different ITS protocol stacks, while the gateway connects different Open Systems Interconnection protocol stacks. Last, the **ITS station border router** is required if a router cannot be used due to different management and security principles.

ITS-S
reference
architecture

One of the main tasks of an ITS is to increase safety in a road system. For this purpose, messages are continuously exchanged between the participants, which are subject to strict requirements with regard to reliability, integrity, and actuality. Two message types are defined for safety-related information, on the one hand, time-based **Cooperative Awareness Messages** (c.f. ETSI EN 302 637-2), on the other hand, event-based **Decentralized Environmental Notification Message** (c.f. ETSI EN 302 637-3). Messages are broadcasted and are subject to certain latency requirements (maximum approx. 100 ms). Their coverage is limited to a certain geographic region (300 m to 20 km). The

Messages
types

size of the payload is capped at 1 kB. There are different types of messages and requirements for efficiency and comfort applications. Their exchange does not take place on the primary control channel (where i.a. safety messages are exchanged) but on service channels that are visited alternatively. This is sufficient for *STRIDE*, since the messages should be up-to-date, but they are not time-critical. Figure 7.1 presents safety messages of both types. An ambulance  can use a Decentralized Environmental Notification Message to request priority, while a RSU 'A' can periodically transmit a warning message, for example, to draw attention to an accident. Furthermore, service messages are distributed, such as traffic light information, where a vehicle  may query the current light phase from a traffic light system .

7.2.1 Requirements Towards V2X Architectures

Vehicular networks rely on the correct transmission and processing of accurate and reliable data in real-time, as manipulation of data by a malicious actor or corruption through network errors could have severe consequences. The dynamic and open nature of V2X architectures stresses security even further, so to allow the secure and safe operation of an ITS, they must meet multiple sometimes contradicting objectives identified in ETSI TR 102 893:

Confidentiality of relayed information is crucial since unauthorized access could lead to the leakage of potentially personally identifiable information. Thus, it has to be ensured that only authorized recipients can obtain information. This extends to communications, management information, identities, and other data sent by or stored in vehicles, as well as RSUs. Furthermore, the location and identity of the ITS participants should remain hidden from attackers attempting to discover this information by analyzing the ITS's V2X communications. Even the analysis of communication flows should not yield any information about the routes taken by users.

Data Integrity is another goal of information security and is of utmost importance in V2X systems, as data manipulation could result in the failure of autonomously driving cars. Thus, communication between ITS-Ss and any information stored within a vehicle or RSU should be guarded against unauthorized and potentially malicious modification or deletion.

Availability of ITS services within the system should not be interrupted by attackers, so there is no delay or disruption when delivering messages to participants under strict time constraints. This is specifically

important for i.a. Cooperative Awareness Messages because they are critical to safety.

Accountability has to be guaranteed for changes to the system, as well as messages sent by ITS participants. In this way, if an unauthorized modification of the system has been performed or a vehicle has sent manipulated data, the ITS can identify malicious nodes and revoke their capabilities. In addition to misbehavior detection, ensuring accountability is critical (e.g. law enforcement) to handling reports sent to law enforcement authorities or insurance companies.

Authenticity is a further integral security objective for V2X networks. All ITS participants must have their own unique identity and role, which also defines their permissions. Impersonating a different identity or acting as an authenticated ITS-S towards other users should be impossible.

All fundamental security goals, namely confidentiality, integrity, and availability, as well as authenticity and accountability to limit communications to trustworthy partners and deal with misbehaving ITS-Ss are consistently considered in V2X security architectures. Further security goals proposed by V2X researchers include non-repudiation [14, 205], consistency, plausibility, and adaptability [269].

Relevant security goals

Security and Privacy Building Blocks

7.2.2

Message integrity and confidentiality are achieved by utilizing end-to-end encryption. Because all authorities and users in an ETSI ITS have a certificate and the corresponding private keys, everyone can establish secure communications. Thus, we assume that outside attackers cannot manipulate the system and describe different inside attackers in the following course.

Resilience against outsiders

To protect the privacy of participants in an ITS, its users should be protected from various attacks, in which an attacker might aim to acquire knowledge of their identity, current location, or starting point and destination. Typically, identity protection can be achieved through anonymity. However, according to the ETSI, anonymity alone is insufficient to protect the privacy of ITS participants, as stressed in ETSI TS 102 941. Furthermore, it is not suitable for use in an ITS, as it conflicts with the requirement of accountability.

Contradicting security goals

Therefore, the preferred approach for providing privacy is pseudonymity as a relieved version of anonymity. A sending ITS-S should never present a unique identity when communicating with other ITS-Ss but instead use a temporary

Pseudonymity in favor of anonymity

pseudonymous identifier. This allows participants to use the services provided by the ITS while keeping their identities hidden, thus preventing attackers from matching their identities with communications in the network. It has to be ensured that pseudonyms are changed frequently and securely to prevent the linkage of one participant's multiple pseudonyms over time. However, it must still be possible for authorities to resolve pseudonymity to identify a misbehaving ITS-S to achieve accountability.

7.2.3 V2X Infrastructure Standards

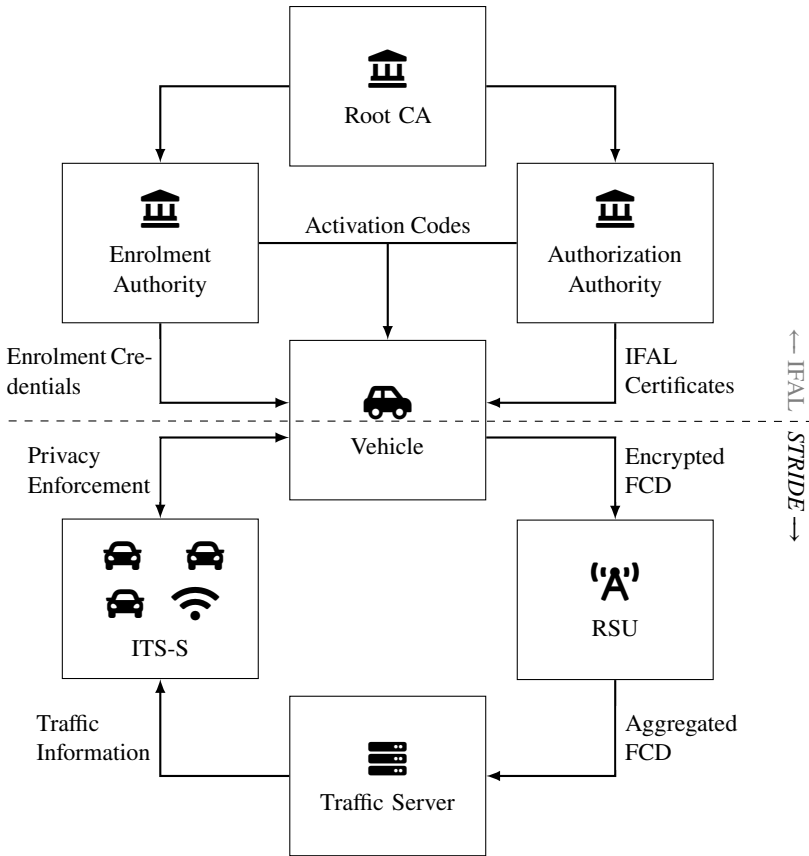
ETSI ts 102
941 We now give a brief overview of the ETSI infrastructure standard, focusing on its trust and privacy management specifications in ITSs as defined in ETSI TS 102 941. Figure 7.2 presents the system architecture with its different entities and relationships, where each entity has access to different data according to its tasks to ensure security and privacy. Since the ETSI's Security Certificate Management System architecture is standardized, it is a foundation for our approach (c.f. ETSI TS 102 940).

ETSI ts 102
940 ETSI TS 102 940 is organized as a PKI whose (single or multiple) Root CA acts as a root of trust and therefore legitimizes the Enrolment Authority (EA) and Authorization Authority (AA) to provide cryptographic material to ITS-Ss. The EA is responsible for controlling the lifecycle of the participating ITS-S. It issues enrollment credentials and can also revoke them in case of misbehavior. The participants then use these enrollment credentials to request pseudonymous authorization certificates from the AA, granting them access to specific ITS services. This separation of knowledge between the EA and AA allows user privacy protection as ITS-Ss only share their unique identifier with the EA, which does not know their short-term certificates actually used for ITS services.

7.2.4 Malicious Behavior in V2X

When developing an ITS architecture, fulfilling the security objectives mentioned requires resistance to several attacks, which can be performed by both outsiders as well as participants of the network.

Classification
of V2X
attacks Ghosal and Conti [143] identified various attacks against V2X-based architectures, which are presented in Figure 7.3 in structured form. Consequently, attacks can be categorized based on the attack's subject with different security objectives threatened that were introduced in Section 7.2.1. First, behavioral attacks target to act on participants found in a V2X environment either for self-beneficial reasons or to execute malicious behavior. Next, attacks targeting hardware and



Components of the ETSI ITS Security Certificate Management System architecture as per ETSI TS 102 941 and the STRIDE extensions. STRIDE uses the existing key management system and includes its functionality as a service with formally defined bounds.

Figure 7.2

software are intended to modify or restrain the ITS-Ss. Infrastructure-targeted attacks include dangerous methods that eventually pose threats to the functioning of V2X technologies. The privacy of participants in particular vehicles may be violated in the context of V2X. Finally, the integrity of submitted packets in a vehicular network is also of interest and thus targeted by attacks aiming to alter or modify them.

Projection of
attacks on
STRIDE

The following is a more comprehensive breakdown of the attacks mentioned, which are of particular concern in the context of *STRIDE*. Consequently, they must be taken into account in the design process.

- ▶ An Intelligent Transport System application may be subject to attacks found in V2X architectures. Attacks aiming at the network's availability include (distributed) Denial-of-Service attacks, which can be achieved by, for example, using flooding or jamming attack. Alternatively, attackers can manipulate packet routing in local network parts by performing black hole or gray hole attacks. The impact of these attacks can be increased if multiple attackers combine their resources or if a single attacker can create multiple identities by employing a Sybil attack.
- ▶ Another attack target is the privacy of network participants. As mentioned in Section 7.2.2, an attacker could attempt to determine their identities, vehicles, and routes. For this purpose, he would employ an identity revealing attack or location tracking. Based on these findings, the corresponding attacker model for this chapter is defined in Section 7.3.5.
- ▶ Many of the communication packets inside a V2X network are sent through its static infrastructure. If an attacker were to take control of parts of the infrastructure, he would gain continuous access to many messages, including authentication and FCD. Accessing these packets and the ability to read, manipulate, or block them would allow him to execute the previous attacks more effectively. An attacker might utilize session hijacking, masquerading, spoofing, or even tampering with the network hardware to compromise its infrastructure.

How different attackers can execute concrete attacks will be shown in Section 7.3.5. The impact of these attacks, depending on different network configuration parameters and traffic conditions, will be evaluated in Section 7.5.

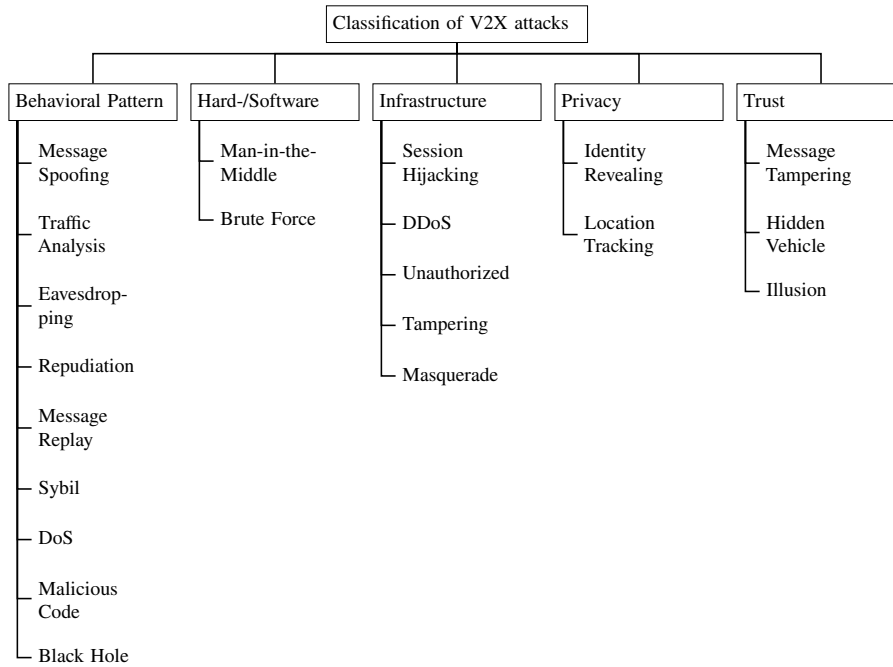


Figure 7.3 Classification of different attacks threatening V2X architectures. (based on Ghosal and Conti [143])

7.3 STRIDE Architecture

We now present the *STRIDE* architecture in detail by explaining how the established requirements are implemented.

7.3.1 Requirements

STRIDE is a Privacy by Design application. Therefore, the architecture must be accessible to the full extent by the participants, while no exceptions regarding privacy are necessary.

Distributed privacy *All information known to a single infrastructure entity should not be sufficient to breach user anonymity*

The first requirement that we formalize is that participants' privacy should remain protected by *k-anonymity*. As long as the different infrastructure organizations and components do not collaborate, deanonymizing and tracking individual users should be impossible. This concept can also be applied to outside attackers, possibly breaching single infrastructure entities.

Data integrity *Data integrity within the system should be ensured by minimizing the impact of attacks.*

Furthermore, *STRIDE* should be resistant to users manipulating or spoofing information. This malicious behavior may be employed to sabotage the network or gain a personal advantage, such as decreasing waiting times at smart traffic lights. In particular, the architecture needs to withstand Sybil attacks and incorrect reporting of traffic information.

Accurate information *The quality of data collection should provide a sufficient level of accuracy for traffic reporting.*

The quality of the data collected and aggregated for processing needs to be considered, as anonymity comes with a trade-off in data accuracy. Therefore, sufficient data quality constitutes the third design goal. The usage of the collected information should be optimized to achieve a traffic prediction accuracy similar to that of current systems.

The achievability of these requirements is the subject of Section 7.3.5 which discusses possible attacks on the framework by each participant while introducing the adversary model.

Privacy and the Integration of IFAL

7.3.2

STRIDE preserves the privacy of participants from outsiders by employing end-to-end encryption. Hence, our focus on privacy protection is on internal attackers. Privacy in V2X networks can be achieved by employing a secure pseudonym scheme, which would require the cooperation of multiple entities to unveil participants' identities and prevent the linkage of different pseudonyms through traffic pattern analysis.

ETSI has published an ITS PKI model for V2X systems (c.f. Section 7.2.3). However, this approach lacks certain features, such as providing certificate batches, which could significantly reduce communication overhead. Instead, the certificate rolling is limited to specific situations, eventually resulting in extended usage and validity of a single temporary identity.

Pseudonymity
as
implemented
in ETSI ts
102 940

The IFAL [379] protocol solves this problem by issuing certificate batches that can only be used for authentication when combined with an activation code for the current time slot. Furthermore, the IFAL pseudonym scheme is compliant with the ETSI infrastructure and uses its EA to periodically distribute the activation codes needed for participation. In this way, misbehaving vehicles can be denied access to the network's services by excluding them from future activation code transmissions, further reducing network overhead. The IFAL protocol also features Sybil resistance by design, as the need for activation codes, when combined with a short rollover period, leads to a maximum number of two valid certificates per participant at any point in time. As this eliminates the need for a dedicated Sybil attack detection protocol, network overhead is reduced even further. The IFAL protocol can be configured by adjusting the validity period $t_{t, val}$ of each temporary pseudonym and the duration of the overlap $t_{t, o}$ between two consecutive pseudonyms (used to eliminate the need for perfectly synchronized clocks).

Issue-First
Active-Later

Components

7.3.3

The *STRIDE* architecture is composed of six components to accurately measure traffic while achieving Privacy by Design, shown in Figure 7.2.

Setup
procedure

According to the IFAL protocol, the EA registers new users and their vehicles and provides them enrollment credentials and a long-term certificate used to authenticate future requests for network services. It is also responsible for the generation of IFAL activation codes required to communicate with the RSUs. Participants use the enrollment credentials to request a set of temporary certificates $\mathcal{Z} = [z_0, \dots, z_n]$ and their corresponding pseudonyms $\mathcal{P} = [p_0, \dots, p_n]$ from the AA that distributes the activation codes generated by the EA.

Concealed
submission
of velocity
information

The RSUs form the physical infrastructure required to measure traffic flow. A vehicle reports its FCD to the nearest RSU once per time slot t_i ² by submitting:

- ▶ The vehicle's current pseudonym p_i
- ▶ The vehicle's current lane l
- ▶ The current time slot t_i
- ▶ The vehicle's current velocity v rounded to the nearest multiple of 5

The message is signed with the vehicle's private key belonging to its currently active certificate z_i to prove its pseudonymous identity p_i . To preserve the privacy of the participants, the velocity to be sent is encrypted with the public key of the Traffic Server (TS) denoted as c . This also ensures the acceptability of participants and reduces the risk of dishonest evaluation of traffic information by an RSU such as, for example, speeding prosecution. We utilize the probabilistic homomorphic encryption scheme number one by Paillier and Pointcheval [289] which works as follows:

- ▶ Generate a standard RSA modulus n as $n = p * q$, where p and q are distinct prime integers
- ▶ Choose $g \in \mathbb{Z}_{n^2}^*$ so that n divides the order of g modulo n^2
- ▶ The public key is $c = (n, g)$
- ▶ The private key is $d = lcm((p - 1), (q - 1))$
- ▶ To encrypt a velocity v , generate a random $r \in \mathbb{Z}_n^*$ and calculate $EncA(c, v) = g^v * r^n \bmod n^2$.

The TS can decrypt all reported velocities v_j per lane l per time slot t_i using its private key d without knowing the reported velocities, as $\prod_j (EncA(c, v_j)) = EncA(c, (\sum_j (v_j)))$ holds.

² Within this chapter, a timestamp t represents a discrete, minute-based time slot. Time slots are fixed intervals based on a global clock provided by the ITS.

Due to the concealed submission of the actual velocity, a single malicious vehicle could report a bogus velocity, significantly skewing the average velocity. Hence, *STRIDE* employs a zero-knowledge proof, adapted from Baudron et al. [38], in which vehicles prove that their encrypted velocity lies within a set of possible velocities:

Validity proof of submitted velocities

- ▶ From the set of allowed velocities $\{v_1, \dots, v_k\}$, a vehicle chooses the velocity v_i closest to its actual velocity and encrypts it to $c = g^{v_i} * r^n \bmod n^2$.
- ▶ It then chooses a random $p \in \mathbb{Z}_n^*$ and randomly selects $k - 1$ values $e_j \in \mathbb{Z}_n$ and $v_j \in \mathbb{Z}_n^* (j \neq i)$ and computes $u_j = \left\{ \begin{array}{ll} p^n \bmod n^2, & \text{for } j = i \\ v_j^n * (g^{m_j}/c)^{e_j} \bmod n^2, & \text{for } j \neq i \end{array} \right\}, j \in \{1, \dots, k\}$ and sends all values u_j to the RSU.
- ▶ The RSU chooses a random challenge $e \in [0, A]$ and sends it to the vehicle (A is any positive integer such that $1/A^t$ is negligible for t iterations of the protocol).
- ▶ The vehicle computes $e_i = e - \sum_{j \neq i} e_j \bmod n$ and $v_i = p * r^{e_i} * g^{e_i/n} \bmod n$ and sends $\{v_j, e_j\}_{j \in \{1, \dots, k\}}$ to the RSU.
- ▶ The RSU tests if $e = \sum_j e_j \bmod n$ and if $v_j^n = u_j * (c/g^{m_j})^{e_j} \bmod n^2 \forall j \in \{1, \dots, k\}$

If a vehicle reports a different, not allowed, velocity, the RSU can report the pseudonym to the AA for misbehavior. Furthermore, all RSUs are interconnected to ensure that each vehicle only reports its FCD to one RSU per time slot, which can be traced by a pseudonym duplicate check by a global RSU provider.

The zero-knowledge proof used is resource intensive w.r.t. channel load, although this is a limited and constrained resource in an ITS. The load can be reduced by decreasing the number of rounds t since it is improbable that an attacker can report a wrong velocity multiple times, and proving his velocity wrong once is enough to permanently exclude him.

Reducing channel load

At the end of each time slot, each RSU forwards the following information to the TS for every lane l :

Deriving global velocity information

- ▶ The lane l
- ▶ The current time slot t_i

- ▶ The number of vehicles reporting their velocity for lane l
- ▶ The aggregated velocity calculated as $\prod_j (\text{EncA}(c, v_j))$

The TS can then decrypt the aggregated velocities with its private key d and use it and the number of vehicles to calculate the mean velocity per lane l . This information can then be used to provide traffic information and, e.g. control smart traffic lights.

7.3.4 Separation of information between entities

STRIDE manages to achieve Privacy by Design by separating information about participants between its different entities. We name the respective information subsequently.

Information known by EA and AA	The EA knows the real identity of each participant \mathcal{I} and a corresponding internal ID shared with the AA for communication purposes. Besides the internal ID, the AA is aware of each vehicle's pseudonyms \mathcal{P} .
Information known by RSU	The RSUs know which pseudonym p_i is at which lane l in each time slot t_i , but they do not know their velocities. To prevent the RSUs from tracking vehicles over time, they change their pseudonym after each validity period of length $t_{r, val}$.
Information known by TS	The TS receives the aggregated velocity and vehicle count per lane, preventing the tracking of single vehicles while still allowing accurate traffic measurement.
Reference to k -anonymity	This model would still allow tracking of individual vehicles in low traffic situations, so vehicles search for other vehicles in the range d_{send} and only send their FCD if at least $k - 1$ other vehicles are present, thus achieving k -anonymity.

7.3.5 Possible attacks on STRIDE

Since all communications employ standard transport layer encryption and entities have to authenticate themselves, outside attackers can not manipulate messages or impersonate participants. Thus, different inside attackers will be reviewed. First, we assume that the Root CA is not compromised. Otherwise, an attacker could impersonate any network authority.

Attacks executable by the EA and AA
If an attacker took control of the EA, he could create unlimited identities, spreading fake FCD. Both EA and AA can also (selectively) stop the forwarding of activation codes necessary to report traffic information and access network services, thus blocking participants' access. We assume that both authorities are honest but curious, i.e., act only as passive attackers.

If the TS acted as a passive attacker, it could try to track single vehicles over time. However, it only receives aggregated velocities of multiple vehicles without any identifiers. As vehicles also only send data when an anonymity group is present, tracking vehicles should not be possible effectively.

Threats posed by the TS

Vehicles in the network can also act as active attackers. By reporting wrong velocities inside the range of allowed values, they manipulate the TS's calculated mean velocity and thus may fake certain traffic conditions, such as congestion. The impact of this attack is evaluated in Section 7.5.2. A vehicle may also send multiple messages with the current pseudonym (in different RSU areas). However, since we assume that all RSUs are managed by a single provider, these distributed components can exchange information. Thus, we introduce a Sybil lookup range ζ_t . It denotes a set of time slots in which messages from the same pseudonym are discarded by RSUs to prevent Sybil attacks.

Vehicle-based attacks

Last, we consider the RSUs as an attacker. Since they independently calculate the aggregated velocities sent to the TS, they could report completely wrong values.

RSUs as a global attacker

Passive attacks Road-Side Units can act as passive attackers and try to track individuals over time. To achieve this, RSUs can analyze all messages sent by vehicles during, e.g., one-day *ex-post*. Since every vehicle will only send once per time slot t_i , the analyzed time frame can be split into discrete time slots. For every observed pseudonym p , there are two possibilities: If pseudonyms are valid longer than one time slot ($t_{i, \text{val}} > \Delta\mathcal{T}$), a pseudonym may have been seen before and the occurrences can be linked. If the pseudonym is seen for the first time in the observed time frame, there are two possibilities. A new vehicle may have just started its journey or entered the ITS-controlled area. Alternatively, it may be a vehicle seen before that has changed its pseudonym and is now using the new pseudonym for the first time. To successfully track vehicles, the RSU has to solve the matching problem between old and new pseudonyms between all consecutive time slots t_i and t_{i+1} .

To analyze the tracking capabilities of RSUs executing this attack, we developed an algorithm that executes such a linking attack. For this purpose, we assume a global passive attacker who has access to all RSUs and analyzes a given time frame *ex-post*. In this scenario, we use a global timer so that the attacker knows when vehicles change pseudonyms. The attack does not attempt to track a specific vehicle over time but, similar to the birthday paradox, searches for random links between old and new pseudonyms. The structure of the algorithm is explained below.

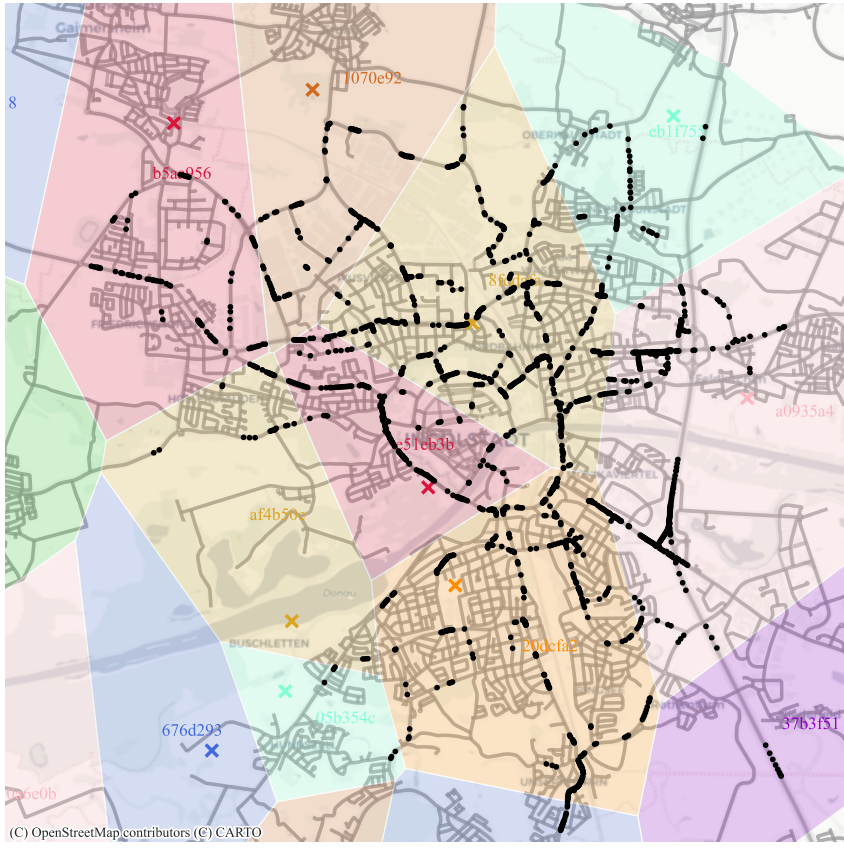
Definition of the adversary model

1. The chosen time frame gets separated into discrete time slots t_0 to t_n of length $\Delta\mathcal{T}$.
2. All recorded pseudonyms are assigned to their corresponding time slots.
3. Starting at time slot t_1 , all possible combinations between previous pseudonyms (t_{i-1}) and new pseudonyms (t_i) are formed as a cross product.
4. These combinations can be filtered, as new pseudonyms must have the age zero, i.e., seen for the first time, to be considered possible successors to the previous ones.
5. The lanes reported by participants in their FCD can be utilized to reduce the number of possible links further. Herefore, the attacker determines the distance between the two reported lanes and decides if that distance can be traveled in one time slot.
6. After this filtering process, the attacker is left with a set of possible successors for every previous pseudonym and has to choose a successor at random.

Active attacks For this work, we assume that RSUs only act as passive attackers. Nevertheless, the RSU provider can be dishonest, attacking not only individuals but also the service itself. We briefly give an outlook on countermeasures in Section 7.6.

7.4 Simulation

A simulation will be used to evaluate and test the system's suitability. This will be done with the help of Sumo, a microscopic simulation environment that allows the trajectories of individual vehicles to be collected and evaluated. A system like the proposed *STRIDE* framework is highly dependent on the behavior of road users, such as the distribution of vehicles on different roads. Using the InTAS [236] data set, a realistic road network can be simulated, whose vehicle frequencies and speeds, as well as driven routes and traffic light circuits, are collected empirically. InTAS describes Ingolstadt in Germany, a city with a high industrial share and shift traffic over 24 hours. Figure 7.4 shows the road network of the given city. The data set also includes different types of vehicles, such as cars and buses.



Vehicles move along predefined, realistic paths in the InTAS road network, which reflects a medium-sized city with industry and old town. Sections of the road network are served by corresponding RSUs, which receive and forward messages from the participants in the catchment area according to the V2X protocol.

Figure 7.4

Table 7.1 Overview of default parameters that are used for the simulation.

Parameter	Values
Sending range d_{send}	100 m, 200 m, ..., 800 m
Sending probability θ_{send}	0.1, 0.5, 1.0
Anonymity set size k	2, 3, ..., 10
IFAL validity $t_{t,val}$	6 min
IFAL overlap $t_{t,o}$	1 min
Attacker proportion κ	0.0, 0.01, 0.1, 0.2, ..., 1.0
Attacker modus ζ	zero, $rnd(0.2)$, $rnd(0.5)$
Sybil lookup time frame ζ_t	1 min

Parameter
settings

We collected FCD for the entire simulation³, i.e. 24 hours, resulting in 179554 vehicles seen and 2195599 FCD entries considering that we only acquired them every 60 seconds. RSU identifiers were added pragmatically. Parameters (and combinations) were changed a posteriori without changing the output to ensure comparability between different parameter settings. (`-device.fcd.period 60` and `-device.fcd.probability 1.0`). This data set is used for further evaluation so that multiple parameter configurations of *STRIDE* are based on the same source data (i.e. routes of vehicles and spawn frequencies) for comparison purposes. Table 7.1 presents an overview of the parameters that were alternated. θ_{send} denotes that a vehicle will send with the given probability within a timestamp. The attack mode ζ defines the behavior of an attacker: *zero* represents an attacker who always sends a velocity of 0 m s^{-1} independent of his situation while the *rnd(▲)* calculates a submitted speed v' as follows: $v' = v * \pm(1 + \blacktriangle)$ for with v being the correct velocity.

Output data

As Listing 7.1 denotes each vehicle has a unique id that it keeps for the whole simulation along with information about its position *loc* (as a tuple of latitude and longitude) and the momentary velocity v in a specific lane l . A lane uniquely describes a road segment on the Ingolstadt map. Within OSM, it is modeled as a way (segment) along with a tag that indicates the number of lanes if applicable. Therefore, it should be noted that a road segment can consist of multiple lanes, e.g. one for each turning direction. Pseudonyms were added according to *STRIDE* specification during evaluation and are not provided by the InTAS data set.

³ <https://sumo.dlr.de/docs/Simulation/Output/FCDOOutput.html>


```

1 <fcd-export>
2   <timestep time="180.00">
3     <vehicle id="carIn101117:1" x="11.442566" y="48.729458"
4       ↵ angle="333.797665" type="default_001" speed="0.000000"
5       ↵ pos="4.898999" lane="-41494424#2_0" slope="0.000000"/>
6     <!-- more vehicles -->
7   </timestep>
8 <!-- more timesteps -->
9 </fcd-export>

```

Example output of FCD data.

Listing 7.1

Thus, we used the following workflow for our evaluation. First, we ran the simulation and then converted the XML data to a `dask`⁴ data frame while adding a RSU identifier to each event crafted by a vehicle. We then adopted different parameter settings and ran the postprocessing. We also combined different settings to elaborate even further, for example, k was combined with d_{send} . The results are shown in the next section. We introduced multiple attack modes to gain information about the reliability of *STRIDE*.

Evaluation
of FPD

STRIDE assumes a specific infrastructure with RSUs distributed throughout the city. We randomly distributed RSUs and created Voronoi regions with each managed by a single RSU. Each vehicle in such a region will send messages only to the RSU responsible for the area. Communication between vehicles and RSUs is done using V2X protocols that are not subject to evaluation. Figure 7.4 illustrates each RSU with its associated zone.

Placement of
RSUs

Evaluation

7.5

Within this section, we will evaluate *STRIDE* based on the simulated data presented in Section 7.4. First, the accuracy of the network utilization reports based on the submitted and encrypted velocity data of the participants is analyzed. The robustness of the system against dishonest vehicles, i.e. the accuracy of traffic reports, is discussed subsequently. Finally, we focus on the privacy aspects of the system, mainly the ability of the global RSU provider to track single vehicles based on the information exchanged according to our attacker model.

⁴ <https://dask.org/>

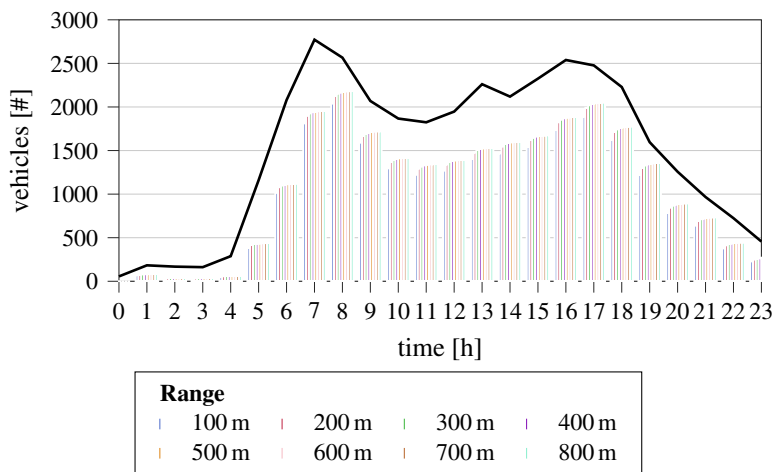


Figure 7.5 Average activity of vehicles per hour throughout the simulation. The results are as seen on a macroscopic perspective (line) and derived by RSUs (bars) for different sending ranges (color) ($k = 5, \theta_{send} = 1$).

7.5.1 Accuracy of the Network Utilization Reports

To obtain an accurate picture of the current street network and its utilization, integer data acquisition is of particular interest.

As a first descriptive and quantitative metric, Figure 7.5 presents the count of active vehicles throughout the simulation period of 24 h (continuous line). It is consistent with a realistic daily schedule where commuter traffic emerges at 6 am. Throughout the day, this remains at roughly a consistent level and decreases toward the end of the day, with a peak at $t = 17$ h. That peak is related to the end of a business day. The bars define vehicles transmitting information according to the *STRIDE* protocol. As a vehicle's sending range d_{send} increases, more messages are received by the RSUs. Also, the number of messages correlates with the number of vehicles that can form a group and also fulfill the condition of k -anonymity.

Apart from the sending range, the chosen value of the privacy level, i.e. the anonymity set size k , is of interest, as it will significantly inflict the sending ability of a vehicle. Recall that a vehicle will only send a message if it reaches at least k other participants. Thus, larger values of k may increase the level of privacy

Relationship
of number of
participants
and sending
ranges

Impact of
anonymity
set size k

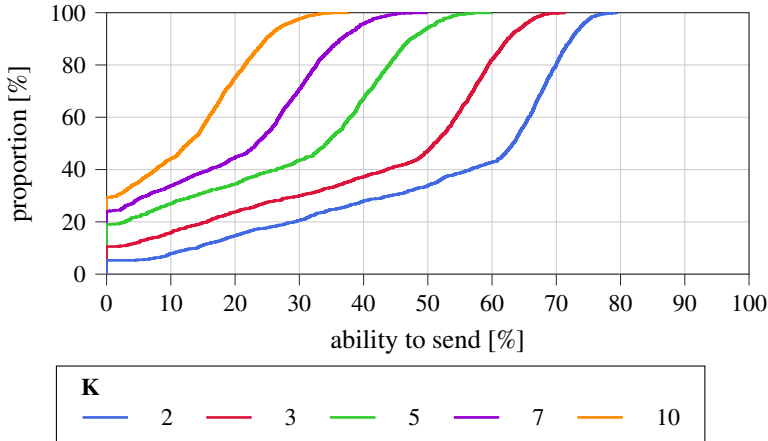


Figure 7.6

Vehicles able to send during the simulation for different levels of k . Higher levels pose stricter constraints on the system, hence limiting the number of successful sending attempts ($\theta_{send} = 1$).

that we will evaluate in the next section. A constant value of $d_{send} = 300$ m was selected, while k was altered according to Table 7.1. The results are depicted in Figure 7.6. With higher values of k , fewer vehicles can reach other participants based on the communication protocol but cannot meet all the requirements to send a message successfully while still ensuring privacy. For example, when comparing $k = 3$ and $k = 10$, the smaller size requirement of the anonymity set allows on average 50 % of all vehicles to send a message with probability of 50 %. Being a strict requirement, $k = 10$ drastically reduces the probability to send message to around 15 %.

This result is reasonable in consideration of Figure 7.7 as it shows that at prime time ($t = 7$ h), on average 11 cars can find each other (solid line) and can satisfy all conditions on k -anonymity which is set to $k = 5$ in this case. With longer sending ranges d_{send} , the probability of successfully forming a velocity group increases, ultimately allowing more vehicles to submit information. The dashed line illustrates the proportion of successful sending attempts compared to those that failed. Higher values are desirable as it means that fewer participants are refrained from submitting information due to privacy requirements. Large numbers of traffic reports allow for the generation of a more accurate picture of current speed. On average for a $d_{send} = 300$ m 70 % of all initiated velocity reports are delivered to a RSU.

Proportion of privacy impact

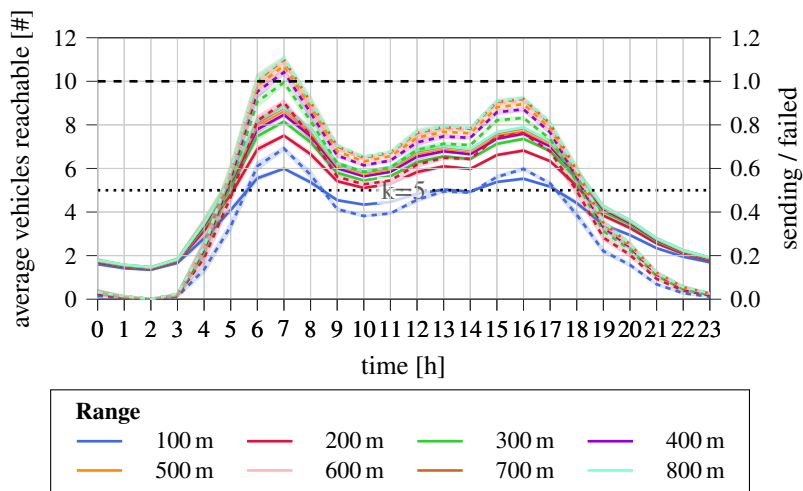


Figure 7.7 Reachability of vehicles of the course of the simulation for different d_{send} . The number of reachable vehicles linearly influences the rate of successful transmission attempts, even if the availability of other participants is only one condition for privacy ($\theta_{send} = 1, k = 5$).

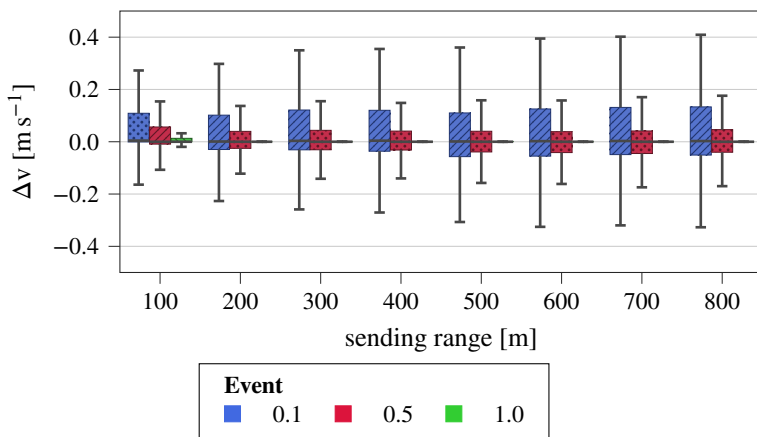


Figure 7.8 Comparison of speed derived by macroscopic view of all active vehicles and based on the TS's aggregated view from traffic reports by vehicles. In summary, the deflection is negligible but varies with the number of included traffic reports ($k = 5$).

However, the number of traffic reports also depends on the number of messages sent, as it inflicts the TS. As we have discussed the privacy requirement is a significant driver of sent reports and the probability of sending the current velocity at a time step θ_{send} is of interest. Sending with each timestamp may improve the accuracy of the data but also stresses the V2X network, with more messages being exchanged on limited bandwidth. Therefore, we evaluated different θ_{send} , that is, every round, every second round, or every tenth round. The last setting is also related to privacy since every pseudonym is used only once for a report. The macroscopically gathered velocity (velocity obtained directly from the simulation, i.e., ground truth) is compared to the velocity inferred by a TS in Figure 7.8. As can be observed, the speed deviates slightly from the ground truth across all sending ranges. The box plots show that the most accurate picture of the street network utilization can be drawn for $\theta_{send} = 1$. Interestingly, the whiskers of the box plots increase with larger sending ranges. The reason may be that the sample for $\theta_{send} = 0.5$ or $\theta_{send} = 0.1$ does not correspond to the true picture. For larger d_{send} , more vehicles can be chosen, and the sample accordingly yields an even more skewed picture. For small d_{send} , the transmission may fail directly because privacy requirements are not satisfied. This limited range of only $\pm 0.4 \text{ m s}^{-1}$ is a result of the high density of cars in such a system. As long as sufficient groups can be created, the transmission range should have no or a negligible effect on the speed consideration. Finally, the speed may be estimated with a high degree of confidence. Thus, *STRIDE* meets the critical accuracy criteria.

Accuracy of
traffic
reports

Estimation Robustness Against Dishonest Vehicles

7.5.2

Vehicles have limited possibilities to manipulate the system, which is shown in Figure 7.9 for alternating percentages of attackers. The graph depicts various manipulation paths. On the one hand, cars transmit an erroneous speed that varies by 20% (red) or 50% (blue) (intended or random due to a system error, for example). In the third situation, the cars send zero values (green). Due to the zero-knowledge proof, the attack's speed range is constrained. As can be seen, the speed deviation in the first two situations may be ignored since speeds diverging upward or downward equalize, whereby they cancel each other out. The third scenario shows that a coordinated attack would certainly provide more significant outcomes. A departure of 1 m s^{-1} from the ground truth may be observed for an assault cluster, including 20% of all vehicles in the simulation. This number increases linearly and reaches a maximum value with 100% attackers. The discrepancy of "only" roughly 3.5 m s^{-1} is due to the fact that the average speed in an urban environment is much less than 14 m s^{-1} (around 50 km h^{-1}).

Falsification
of velocity
data

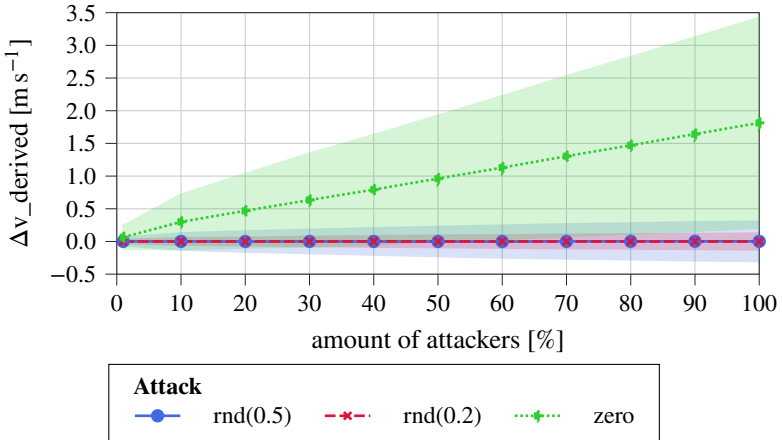


Figure 7.9 Divergence of speed for different ζ and amounts of attacking vehicles. The average deflection is acceptable within the given scenario ($d_{send} = 300$ m, $\theta_{send} = 1$, $k = 5$).

7.5.3 Location Privacy-preservation Against Dishonest RSUs

Pseudonym rollover example

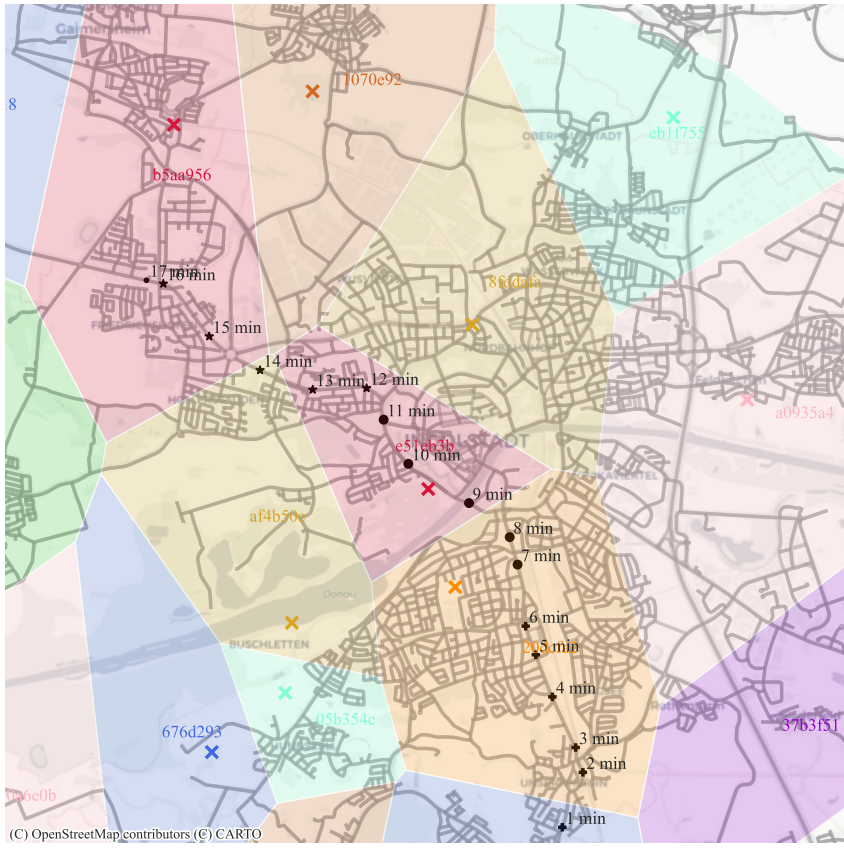
According to the attacker model defined, the RSU operator attempts to link the pseudonyms to monitor distinct individuals. The trace of a single vehicle with the pseudonym employed represented as a unique shape is illustrated in Figure 7.10. IFAL validity ($t_{i, val}$) is set to 6 min with an overlap of $t_{i, o} = 1$ min. Hence, the first pseudonym change will occur at $t = 7$ min. The attacker now wants to link either pseudonyms p_+ and p_\bullet or p_\bullet and p_* because that enables him to track the route of the given vehicle.

Linkability of successive pseudonyms

The attacker's ability to perform the linking process is depicted in Table 7.2 for different sending ranges and lower bounds for the anonymity set size. Increased k values should reduce the likelihood of proper linking. The same holds for higher transmission ranges that should increase the number of vehicles eligible for the process. Both hypotheses have been proven. Except for sending ranges of 100 m or $k \leq 5$, it is invariably possible to concatenate two pseudonyms with a probability of less than 10% if the attacker has created a group in which the actual vehicle is placed. These values are determined throughout the simulation and hence, for low traffic densities.

Selectability of correct vehicle

The previous analysis was based on the fact that an attacker is capable of selecting the correct candidate from a set of candidates if the true candidate is within it.



Trajectory of a single vehicle illustrating the pseudonym change every $t_{l, val}$. In addition to the pseudonym change, the vehicle passes different RSU areas, although, pseudonym changes are time-based instead of location-based. Figure 7.10

However, the attacker is seldom able to even find a set with the true candidate being part of it. It became clear that with a greater sk , the attacker would be unable to even choose the proper car as a prospective candidate after changing the pseudonym due to the sheer number of vehicles and groups. This is achievable in less than 5% of situations where $k \leq 5$. Bear in mind that the total number of cars in the simulation is less than 200. In none of the instances analyzed did the attacker create a bogus link, i.e. he never falsely suspects a participant. Furthermore, we evaluated another time step ($t = 936$ min) with approximately 10 times the number of vehicles. However, we may infer that an attacker's capacity to correlate two consecutive pseudonyms without targeting a single vehicle remains implausible. With increasing traffic volume, accuracy drops to fractions of a percent.

In-depth Analysis of Pseudonym Changes

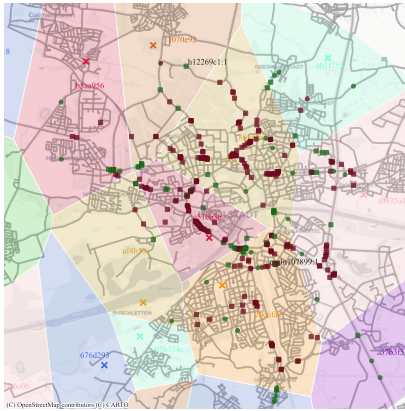
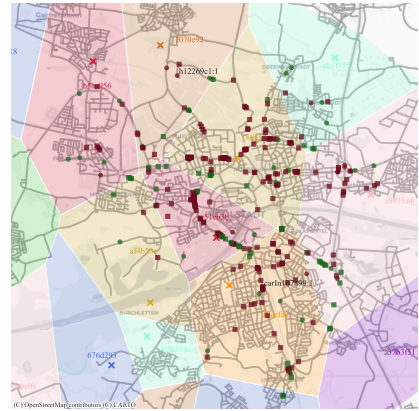
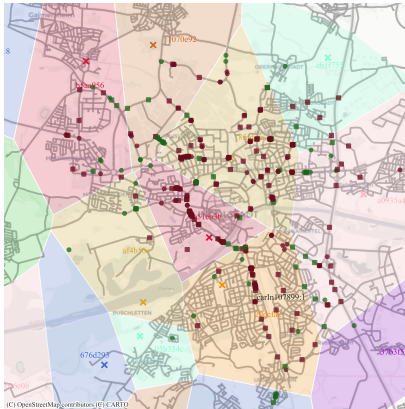
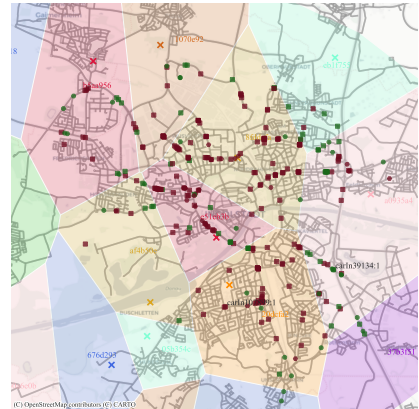
The state of the simulation is depicted for four different timestamps in Figure 7.11. All the vehicles and the RSU areas are shown. The shape represents the usage of a pseudonym, while the color illustrates whether a vehicle was previously seen by a RSU.

A pseudonym can either be new (in the case a vehicle enters the street network), changed (if a certificate change is permitted and performed), or reused (if the vehicle sends a traffic report again with the same pseudonym). A vehicle can be new to a RSU (if it sees a traffic report for the first time from a pseudonym) or known (if the same pseudonym sent at least two traffic reports).

Observed
simulation
states

At each time step (c.f. Figures 7.11a to 7.11d), pseudonyms of the three states are present. Due to the design of *STRIDE* and the chosen parameters, in particular $t_{l, val} = 6$ min in combination with $\theta_{send} = 1.0$, most pseudonyms are reused from the previous timestamp. Only vehicles that transition between RSU areas are those that first contact a specific RSU. New and changed pseudonyms occur at each step, making it hard for an attacker to decide if the vehicle is initially reporting information or has just changed its pseudonym. For example, vehicle `h12269c1:1` was reusing its pseudonym in $t = 10$ min but changing it in $t = 11$ min. Moreover, it stayed in the same RSU area and sent two successive velocity reports. Due to the activity in that area, new vehicles enter the area in $t = 11$ min. Two have also changed their pseudonym simultaneously in the same area as the vehicle in focus did, making it hard for an attacker to differentiate between them, even if location-based interference may be used. The vehicle disappears in $t = 12$ min. In $t = 13$ min, a new car appears in a group of other vehicles (`carIn39134:1`). Also, at the same time, at the last visible spot of the

Historical
and location-
based
attacker
knowledge

a) State at $t = 10$ min.b) State at $t = 11$ min.c) State at $t = 12$ min.d) State at $t = 13$ min.
Symbols

●	×	♦
■	■	

● New p × Same p ♦ Changed p
 ■ New RSU ■ Reused RSU

Figure 7.11 Vehicles moving within the street network with specific vehicles highlighted. Shapes represent the different types of the pseudonym with all types present at every time step implying an ongoing pseudonym overroll. The communication history with an RSU is indicated by the color ($k = 5$, $d_{send} = 300$ m).

vehicle h12269c1 : 1, a car appears initially. However, both cars are not the same, but participants with different I s, which might not be evident for an attacker if he tries to use historical knowledge.

Conclusion

7.6

In this chapter, we presented *STRIDE* that enables privacy-friendly submission of real-time velocity reports to a centralized instance that is eventually used to assess the current utilization of the network. According to the Privacy by Design paradigm, all functionality is included in the protocol itself.

STRIDE is based on a common V2X infrastructure to quickly enable adoption and uses an existing and standardized pseudonym scheme called IFAL to balance integrity and privacy. Therefore, participants include vehicles that provide speed information and are interested in having their location trajectory protected, RSUs relaying data, and a TS providing the traffic data service. Vehicles are solely responsible for their privacy by allowing them to change their pseudonym more frequently or to refrain from sending traffic reports. Privacy is implemented in the system by relying on k -anonymity. This results in a different interpretation of Privacy by Design since vehicles may not be able to provide FCD if the privacy requirements (i.a. greater than k vehicles reachable) are not met. Hence, a vehicle may not participate. Since this protective measure is ultimately mingled into the system design (instead of an external PET), we still consider *STRIDE* to be a Privacy by Design architecture. The proposed architecture uses RSUs already present in V2X environments to collect and forward encrypted information from vehicles to a TS that uses homomorphic encryption to find the average speed of the roads in a specific street network.

Architectural
summary

Our evaluation shows that *STRIDE* can protect against a global attacker, in particular an RSU provider who knows of the activity of all participants in a street network. There is a negligible chance that he will use the sparse information available to link two successive pseudonyms to track a vehicle in the end. We provided an in-depth analysis of pseudonym rollovers to understand the efficiency of this privacy building block. *STRIDE* archives to mitigate such passive attacks with up to 100% probability. The evaluation is based on real-world traffic data to simulate realistic driving paths and speeds. Using the proposed architecture, a Traffic Server can derive accurate traffic reports with almost zero divergences between the actual macroscopic speed for lanes and the one derived via privately submitted vehicle data.

Evaluation

- Limitations** The current implementation does not allow vehicles to transmit data once a sufficient set of cooperating partners is unavailable. Generating dummy messages is also not an adequate workaround due to Sybil resilience; at any point, each vehicle is only allowed to be active with one pseudonym (except for the transition phase $t_{t,o}$). At the same time, no messages can be generated by RSU since its intention, in terms of the attacker model, is to track individuals, and consequently, it is not trustworthy to be a supporting entity for privacy. At the same time, these low-traffic situations represent a new attack vector since an attacker can exclude inactive zones or vehicles originating from inactive zones from his candidate sets; after all, he could not know their pseudonym. In the context of the simulation, such situations rarely or never occurred and were irrelevant depending on the observation area. However, due to this limitation, *STRIDE* is somewhat less suitable for rural areas, whereby it should be noted that there the speed estimation should correspond to the guideline speed due to lack of traffic.
- Outlook** For future work, it is meaningful to further stress-test the resilience of *STRIDE* by introducing a dishonest attacker, namely the global RSU provider: RSUs deliberately manipulate their stated velocities in this case. One way to mitigate these attacks is to require RSUs to keep signed FCD messages received from vehicles, which could be tested using variable controls relying on blockchain-based approaches. Additionally, vehicles may broadcast their FCD to enable monitoring of the RSUs' forwarding behavior. Furthermore, it might be meaningful to analyze different pseudonym switching schemes in addition to the time-based method proposed by the IFAL protocol, such as switching at the RSU area borders, similar to the concept of mix zones [42].

Chapter 7 focuses on a V2X system with interconnected entities, including vehicles or traffic lights. In such a scenario, the information can be used to warn other participants or optimize traffic flow. For example, a traffic light can count the number of approaching vehicles and derive the best light pattern sent to vehicles [323, 325]. The vehicles can then change speed to get to a green light. In addition, participants gain information on road conditions (e.g. if it is slippery due to rain). Every participant contributes to the Intelligent Transport System and collects data to share with others for the greater good. Therefore, entities can be considered mobile sensors. Consequently, we use the term Everyone-as-a-Sensor (EaaS) which is a combination of crowdsourcing and opportunistic sensing (c.f. Section 1.1).

Although ITSs are not widely implemented, modern vehicles are yet to rely on recent, accurate and comprehensive information. ADAS need the structure and conditions of the road to safely adjust related properties such as speed, distance to the vehicle in front, or turning speed. Today, however, information is collected differently. For example, in OpenStreetMap (see Section 5.5) volunteers collect data and update the map material accordingly. To relinquish from that paradigm that information is explicitly collected rather than implicitly compared to an ITS scenario approaching a more direct adaptation of ubiquitous computing, we propose a system to approach the paradigm of Everyone-as-a-Sensor and thus massively increase the crowd of volunteers.

Crowdsensing and crowdsourcing

This system is called *ROADR*. It is a web-based platform and an Android application that can be easily retrofitted. With the help of this system, the structure of a road network can be automatically captured, and semantic information can be collected using an opportunistic sensing approach. For example, when new

ROADR

Acknowledgement

Parts of the research presented in this chapter are based on work published previously [317, 319, 326] and supervised work [S3, S13].

traffic elements are detected, they can be automatically added to the OSM map. Since an individual's movement data is particularly sensitive, smartphone sensor data analysis is performed in the user's domain by optimizing machine learning models for mobile devices and applying them there. W.r.t. the Privacy by Design paradigm introduced in the previous chapter only aggregated information without reference to individuals will leave a device.

Contribution This chapter depicts a holistic view of the crowdsensing and crowdsourcing scenario in a mobile environment using the Everyone-as-a-Sensor concept. In fact, we present

- ▶ an Privacy by Design-based architecture to collect and process mobile, privacy-sensitive sensor data,
- ▶ an Android demonstrator that collects, processes, and evaluates sensor data using efficient Machine Learning and sophisticated sensor fusion.
- ▶ a thorough and in-depth discussion of sensors derivable from sensor data, including a description of biases and environmental constraints and impacts, and
- ▶ a real-world evaluation of the *ROADR* platform including event detection accuracy and crowdsourcing benefit analysis.

Structure We first present related work in Section 8.1, before conceptually and technically introducing the holistic, privacy-aware *ROADR* platform in Section 8.2 [317, 326]. It also analyzes meaningful sensor data w.r.t. the map and road network scenario. All required information is collected solely by a smartphone app that is also thoroughly presented in Section 8.3 [326]. Then, we show in Section 8.4 that FPD data allows deriving information about traffic circles [319], traffic lights [S13], and road works [S3]. Approaches to detect said events are discussed in Section 8.5 that are implemented in the mobile companion and remote platform. We evaluate our platform using the known real-world data set in Section 8.6. Section 8.7 concludes this chapter.

8.1 Related Work

Location-aware sensor processing The wide adoption and presence of sensors in smartphones motivated research to use the gathered data to obtain syntactic and semantic information about a road network. The given research field is well focused on academia and industry, and research is conducted on a wide range of applications. In most cases, information

from an IMU in combination with GPS is used first to leverage the sensor data and then map it to a location. *ROADR* also tries to generate added value for the existing map material and, ultimately, users. Consequently, smartphones lower the barrier to having data collected by a crowd of people by omitting the requirement for specialized hardware. Therefore, they help to reduce the imbalance of users and contributors that is prevalent in OSM, for example.

Applications or underlying platforms can be differentiated based on the approach to data processing. For example, one group of the proposals found processes the data exclusively on the device and transmits aggregated results to a typically cloud-based platform. In contrast, the second group uses mobile devices among users purely for data collection (e.g. [17, 158, 169, 256]). Raw data transmitted is subsequently processed online. Some works also apply truth-discovery algorithms [158] to further enhance the quality and accuracy of error-prone sensor data from mobile devices. Data is also combined with external services [158]. Hybrid methods include both approaches [139].

Types of
data
processing
platforms

Often found are works that attempt to detect road quality [158, 347] or related features such as potholes [425] or speed bumps [139]. This is often done using the accelerometer by looking at one standard deviation (e.g. [347]). The demographic influence in collecting the test data should not be neglected in such work. Comparability or transfer of approaches is often not trivial (compare developed road systems with emerging countries). Furthermore, work can be found that focuses on the detection of road elements [17, 319]. Aly et al. [17] addresses passers-by in addition to vehicles to detect a variety of elements (i.a. tunnels, crosswalks, bridges). Traffic circle detection using local ML models is the focus of Roth et al. [319], whose proposal operates on sensor data from the accelerometer and gyroscope. Of further interest is the detection of traffic lights, where typically image recognition-based methods using the camera are applied [235].

Common
evaluation
types

However, some proposals exclusively provide a platform-agnostic application suitable for processing sensor data in a crowdsourcing manner [63, 112], some with a focus on privacy [317]. Other works aim to increase the usefulness of maps by adding points of interest [329] or making changes using elements of gamification through simple tools¹.

Crowdsourc-
ing of sensor
data

In contrast to other work, *ROADR* is developed using the Privacy by Design paradigm and considers user privacy throughout. We deliberately do not forward raw sensor data because it is susceptible to side-channel attacks to identify [423]

Our
approach

¹e.g. <https://maproulette.org/>

or track (see Chapter 12) subjects. Many works do not protect the user against such attacks (e.g. [17, 158, 169, 256, 347]). Moreover, we do not limit ourselves to a single objective; instead, the *ROADR* platform offers a holistic approach to data collection and automatic integration into maps such as OSM. Finally, the *ROADR* mobile companion tries to minimize the impact on the user by minimizing battery usage, effortless workflow, and transparency.

8.2 ROADR Platform

Overview
and
requirements

The *ROADR* platform considers vehicles as mobile sensors that move along predefined paths (streets) in a network. The data basis of the network is OSM, introduced in Section 5.5. As shown in Chapters 2 and 4, the FCD that are the subject of this study can also be collected from a smartphone (FPD) of similar quality, given certain factors. A crucial factor is that the *ROADR* app (see Section 8.3) records the data accordingly. Measurements \mathcal{M} , as defined, are recorded continuously. Furthermore, it is assumed that the mobile device is not moved only to a minimal extent while driving. In this case, sensor data enables inferences about a vehicle's passed trajectory (c.f. Chapter 4 and Section 5.4) in the meaning of crowdsensing (c.f. Section 1.1). Data is gathered and analyzed using Machine Learning by the local *ROADR* application without the need for a remote connection. Subsequently, if the app detects some elements of interest (events), they are forwarded to a gathering instance to provide additional value to the community.

8.2.1 Goals and Requirements

These inferences should be aggregated, interpreted, and optimized accordingly by the *ROADR* platform. The intention is to return the syntactic and semantic information of the road network in terms of global knowledge. Therefore, the platform aims to automatically clean up errors and inaccuracies in existing map material and store additional information such as the average waiting time at a traffic light in the form of meta-information in OpenStreetMap. Specific requirements are placed on the crowdsensing platform:

Limited Accuracy The platform should optimize the map data, which requires a reliable data source with a low false-positive rate. However, the mobile sensor data itself is error-prone as its accuracy can vary from device to device, as we have illustrated. Furthermore, it might

be affected by external factors such as driving behavior. In our crowdsourcing application, a threshold mechanism will be implemented in which the crowd ultimately verifies the claims of individual participants and appropriately manages error-prone sensor data.

Privacy Trajectories may enable keen insights into a user’s everyday life. Users should reveal as little personal data as possible while using crowdsensing apps. As a result, a necessary condition is that the data cannot be connected to a particular topic.

Ease of use To achieve market penetration, the entry barrier must be kept to a minimum. Systems that need a lengthy setup or the installation of additional hardware are ruled out. It is critical for the platform’s success that consumers may engage through their smartphones in a used-to way. Therefore, a smartphone application lends itself to.

Architecture

8.2.2

The *ROADR* platform is built on two pillars and includes four stakeholders. It addresses the requirements named in Chapter 6. The architecture is depicted in Figure 8.1.

First, vehicles are mobile sensors that sense events by analyzing the sensor data stream. The user side contains a data layer that queries the sensors to forward it to a detection engine that interprets i.a. CEP and ML to identify events accordingly. In addition, this domain includes a management layer for credential and model administration. The second pillar is the remote Centralized Data Processor (CDP). It is responsible for processing the events of the mobile sensors accordingly. The CDP consists of four layers that perform different tasks. Individual events are aggregated by the aggregation layer under certain conditions (similar location and event type). Subsequently, semantic and syntactic information is processed in the learning layer on the basis of the events. For example, the average waiting time of a traffic light can be calculated hereafter the direction of travel and the waiting time have been extracted from an event. The service layer can pass on the findings accordingly and is designed dynamically. A third stakeholder is an issuer, who provides pseudonyms to vehicles to enable them to submit events. Additional stakeholders are external services, for example, OSM.

Distributed processing

The detection engine ensures privacy by processing privacy-critical sensor data (c.f. [317]) on the user’s device. Thus, only intentional information, known in the context of this work as events, is sent. At the same time, this results in a scalable and robust platform, as the centralized infrastructure is lifted from the heavy tasks

Privacy-aware data processing

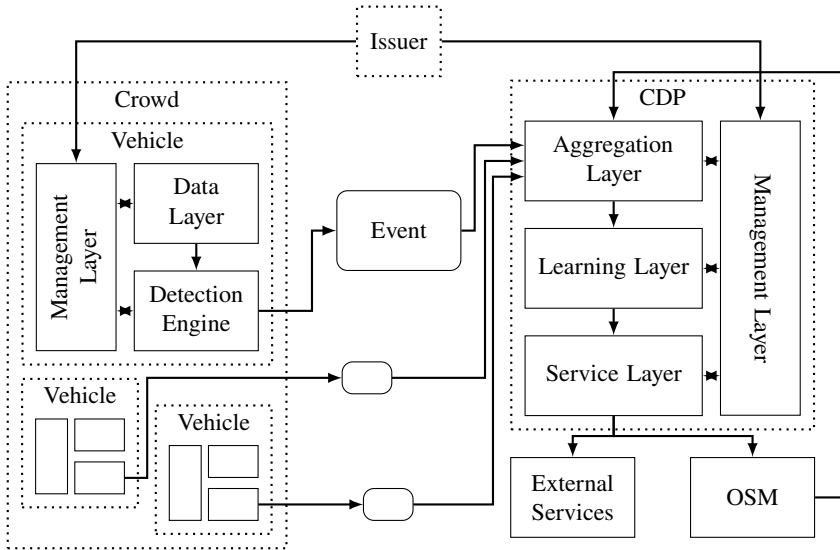


Figure 8.1 The ROADR platform is built on two pillars and four stakeholders. First, vehicles are the mobile sensors to sense events by analyzing the sensor data stream using Complex Event Processing (CEP). If an event is found, it is forwarded to the Centralized Data Processor (CDP) that processes it accordingly (e.g. persisting and aggregating it). A third stakeholder is an issuer, who provides pseudonyms to vehicles to enable them to submit events. Additional stakeholders are external services, for example, OSM.

of data processing and analyzing. It is left with only sanity checks, aggregation processes, and persistence operations. Due to the application's smartphone-based nature, a rapid adaption rate and a large amount of data may be expected. To further support adoption, gamification can be included on the platform [317]: For each confirmed event, a user can collect points that underlie his support of the community. However, some limitations must be addressed, the most significant of which is battery usage. Additionally, all machine learning models are designed for accuracy since missing an event is less critical than incorrect information discovered over time by the community. Models included in the mobile application can be updated through a feedback channel provided by the CDP.

Challenges One challenge in designing such applications is the balance between integrity and privacy. The privacy requirements of the users must be taken into account. For example, it should not be possible to trace users' trajectories. This is

possible if submissions of individual persons can be linked, which is prevented by ensuring that only aggregated data leaves a user's device. Additionally, no identifiers or personal characteristics can be found. Integrity stands in contrast to this requirement. Thus, the data must be correct to provide added value. To protect against fraud, such as Sybil attacks (see Section 7.2.4), ABC4Trust (c.f. Sidebar D) is used. ABC4Trust is implemented in the management layer of the platform, which is present in both of its components. The PoC abstracts this component.

Components of ABC4Trust

Sidebar D

Rannenberget al. [307] is a privacy-enhanced attribute-based credential (privacy-ABC) system that was funded by the European Union. It enables the development of trustworthy applications that integrate seemingly incompatible objectives such as reliability, integrity, and privacy. A typical design abstracts away the particular implementation of the ABC system, allowing developers to create sophisticated but secure applications.

ABC4Trust specifies five distinct roles, which are as follows: *User*, *Verifier*, *Issuer*, *Inspector* and *Revocator*. Additionally, it defines `credentials` as containers for `attributes` specified by either the user or the issuer, confirmed (blindly) by an issuer (e.g., a network provider using the device's SIM card), and held by the user. Possessing and knowing a signed credential enables access to a remote system secured by a Verifier, in this instance, the centralized data processor. As a result, the credential enables the submission of new events. Furthermore, a *pseudonym* is a user's (temporary) identity that enables (limited) linkability, if necessary, which is critical for an integer platform. Both components may be associated with a user-only secret, providing another degree of validity.

In *ROADR*, we utilize this functionality to protect against assaults such as flooding by segmenting the map into parts where each vehicle ultimately generates the same pseudonym each time it reports an event discovered in this sector. ABC4Trust is implemented in the management layer of both platform components.

8.2.3 Aggregation of Events

Contradict-
ing
information

In the context of a crowdsourcing solution, events must be recognized and reported by a large number of participants. In order to generate knowledge from this, the information must be aggregated accordingly. Therefore, several challenges arise in the location-based topic. First, the inaccurate location information must be taken into account, as vehicles can move very fast. Consequently, it can be assumed that two vehicles, which can describe the same situation, do not provide identical location information. Also, the time dependency has to be considered, i.e. some events may be temporary. As an example, road works occur and are finished; hence, the platform has to handle such scenarios. This is particularly challenging, as road works are only detected in the present case but are undetected if they do not exist anymore. Consequently, no events are submitted for road work. Therefore, this platform needs to verify if a road work has been confirmed from time to time.

8.3 Mobile Companion

In this section, the accompanying Android application from *ROADR* is presented. It performs two basic tasks. First, the application collects data and preprocesses it. Second, the prepared data is used for the detection of various traffic events, which will be introduced in the next Section 8.4. When any of the events are detected, they are shown transparently on a map. This underlines the claim to trust being a core building principle of *ROADR*. Also, the user interface shows an assessment of the current road quality. The user interface is shown in Figure 8.2

Modular
data
processing

The data processing architecture of *ROADR* is depicted in Figure 8.3. Due to its modular design, the application can be expanded accordingly and enables the processing of one data stream for the detection of several events at the same time. We now explain each step of the CEP-based architecture in more detail.

8.3.1 Collection and Preprocessing

Data is collected from the built-in sensors on an Android device using the official API. We then perform multiple attribute sensor fusion as well as time-based fusion (c.f. Sidebar C).

Data fusion

We use ReactiveX² to observe sensor events asynchronously without blocking other operations continuously. The applied technique is also called *observer-*

²<http://reactivex.io/>



User-Interface of the ROADR application. The user interface displays multiple detected event types including their location. There are some basic controls to start and stop the detection.

Figure 8.2

pattern. In particular, the raw values of the accelerometer, the gyroscope, and the GPS are continuously sampled. The sampling rate, i.e. frequency f varies between the sensors, as explained in Section 5.2. Readings from a sensor x_t at time t' are integrated into a single measurement m for time t by averaging values $\{x_{t'} \mid t - f^{-1} \leq t' \leq t\}$. Thus, errors such as outliers or noise can be reduced accordingly (c.f. Section 2.3) to allow processing of the data. Since the GPS in Android is rather restricted to $f = 1$ the last *loc* is used. As previously explained, this yields an inaccuracy in the location submitted with detected events, although our platform is capable of aligning such events to a correct location using crowdsensing techniques.

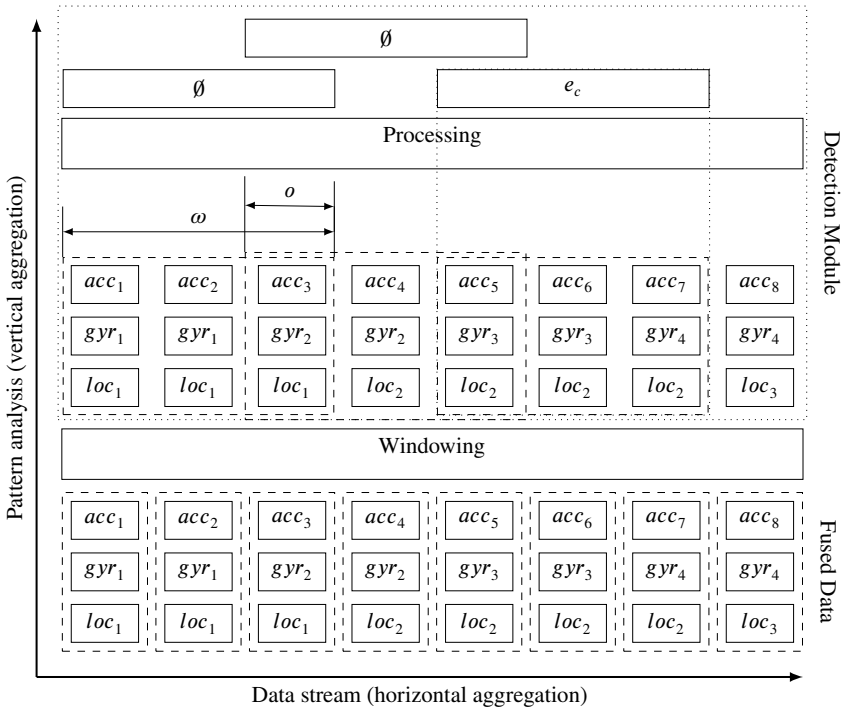


Figure 8.3 Multi-level workflow within ROADR. The CEP process is based on the fused data as shown in Figure 5.5. A detection module performs windowing using module-specific parameters and pattern analysis to find sophisticated events.

Publication
of new mea-
surements

Once a new measurement is generated every f^{-1} seconds, it is published in the respective detection modules that perform further processing to detect events. Each measurement is forwarded to all subscribed detection modules simultaneously to perform the computation in parallel.

8.3.2 Windowing

Generation
of windows

Windowing models are employed to process the continuous stream of measurements to assess for events. Let $W(\widetilde{\mathcal{M}}, a, \omega)$ denote a function to generate a *Sliding Window* $\widetilde{\mathcal{M}}'$ of fixed length ω starting at the marker a on a continuous and growing data stream of measurements $\widetilde{\mathcal{M}}$. Sliding windows are usually applied to reduce the risk of missing patterns in time series [270]. $\widetilde{\mathcal{M}}$ is the sequence of

measurements received via subscription, while $\widetilde{\mathcal{M}}' \subset \widetilde{\mathcal{M}}$, hence W is a surjective. Windows are then used to calculate specific explanation features such as statistical attributes (e.g. using min, max, mean, std). All elements in a sliding window are of equal importance in contrast to other approaches such as the *Damped Window Model* [244].

$\widetilde{\mathcal{M}}'$ only holds measurements whose marker a lies in the interval $[a - \omega + 1, a]$. Other measurements are discarded. The elements of the window are therefore defined by an arbitrarily placed marker, which may be of a specific type b . Therefore, W may use an additional mapping function M_W to map ms to that specific type b to validate if it is part of that specific interval. Different approaches can be used to define M_W and thus modify the construction of a sliding window. \mathcal{M} are time series-based trajectories that are location-aware. Hence we define two mapping function, one to map measurements to a timestamp $M_{W_t} : \mathcal{M} \rightarrow \mathcal{M}_{*,t}$ and one to derive distances from a sequence of measurements $M_{W_d} : \mathcal{M} \rightarrow D$.³ For instance, if t_{last} is the current timestamp of the last measurement, all older objects other than ω are removed.

Different window selection strategies

In addition, *ROADR* uses overlapping windows. Overlapping windows are further defined by an additional overlap parameter called o . The parameter describes the amount needed to move the marker a forward along the sequence \mathcal{M} . In the easiest case ($o = 0$) there is no overlap for two successive windows, hence: $W(\widetilde{\mathcal{M}}, a, \omega) \cap W(\widetilde{\mathcal{M}}, a + \omega, \omega) = \emptyset$. This is not desirable in some cases because e.g. a traffic circle pattern could be separated in unfortunate positions and may be hard to detect. Overlapping windows reduces the risk of missing a pattern by setting the marker of the subsequent window to $a' = a + \omega * (1 - o)$. For example, if o is 0.2, both windows overlap 20 %.

Overlapping windows

Detection

8.3.3

A detection module is a modular and self-contained package for recognizing a specific event. Its input is the continuous data stream and windows, respectively, while the output is potentially found events that are also published similarly compared to measurements. There are no additional dependencies between any module or other parts of the application. Each detection module can define its own parameters, including window size and overlap, to fully capture the corresponding patterns in one window. *ROADR* in its current state has four modules to detect traffic circles, traffic lights, road works, and assess road quality.

Atomically tasked modules

³ $\mathcal{D} = [\text{Dist}(m_i, m_{i+1}) \mid i \in (0, \dots, |\mathcal{M}| - 1)]$ with Dist returning the geographical distance of two measurements $m_{loc,i}$ and $m_{loc,i+1}$.

As the stream of measurements is constant, detection modules are built in an efficient way: Each module performs sanity and precondition checks to fast-fail if the probability for an event is too low. We will describe the methods applied when explaining a specific detection module in Section 8.5.

8.4 Overview of Events

As we have already shown in Section 5.4, a sequence of measurements \mathcal{M} can give insight into the course of a journey in order to make general statements about the driver or the environment. For example, the presence of many curves implies a curvy road, which is potentially less common in cities but more common in rural areas. Generally, predictions based on sensor data should be precise and accurate, but this is often not possible due to many disturbance variables. Nevertheless, there are salient structures in sensor data that are worth considering, as they allow insights into the structure of a road network.

Selected elements

In this section, four events are presented in detail, which may be processed by the *ROADR* platform to correct the map material and thus increase the actuality and accuracy, but also to complete the map material. They are particularly crucial for ADAS or similar systems. For instance, the presence of a traffic circle directly impacts the travel speed, which a cruise control may set.

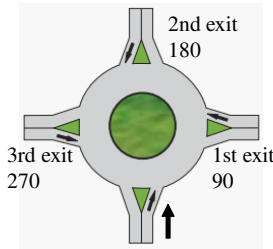
8.4.1 Traffic Circles

Definition

A traffic circle is a type of intersection where traffic flows in a circle around an island in one direction. Unlike an intersection, traffic is usually prioritized so that vehicles in the traffic circle have priority. To ensure consistency in nomenclature, the conventional structure of a traffic circle as defined by the German Automobile Club is used [370]. It is shown in Figure 8.4. The first exit is about 90° , the second exit is approximately 180° , and the third and typically final exit of a traffic circle is approximately 270° . Simply speaking, it is assumed that an exit may deviate up to 45° . However, remember that traffic circles may have varying numbers and exit placements.

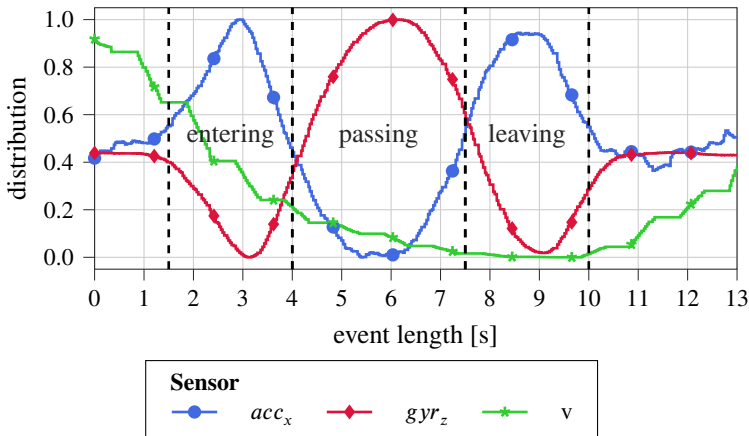
Rough pattern definition

We empirically determined that the traffic circles show a unique pattern on the gyroscope and accelerometer. Especially on the gyroscope's z-axis (gyr_z), the accelerometer along the x-axis (acc_x), and the velocity v . Figure 8.5 illustrates the course of the sensor readings of entering, passing through the traffic circle, and then leaving it at the second exit. The speed decreases before entering a traffic circle, remains constant once within, and increases again once exiting.



Exemplary traffic circle with four exits. (based on Tober et al. [370])

Figure 8.4



Pattern for traffic circle exit at 180° (2nd).

Figure 8.5

While the trajectory of gyr_z takes the shape of a “W”, the acc_x is opposite and resembles an “M”. Since acc_x reflects centrifugal forces, which are mainly directed in the opposite direction compared to steering maneuvers, both sensors describe the same issue; hence, both can be used to confirm an event (i.e. sensor fusion). These patterns result from three phases of a trip through a traffic circle: entering, circumnavigating, and leaving. Sensor measurements are normalized to fall in [0, 1].

Differentiating between different exits is performed by observing the phase “passing” within the traffic circle, which is lengthened when exiting by a different exit. A later exit from a traffic circle results in a prolonged interval within the

Detection of different exits

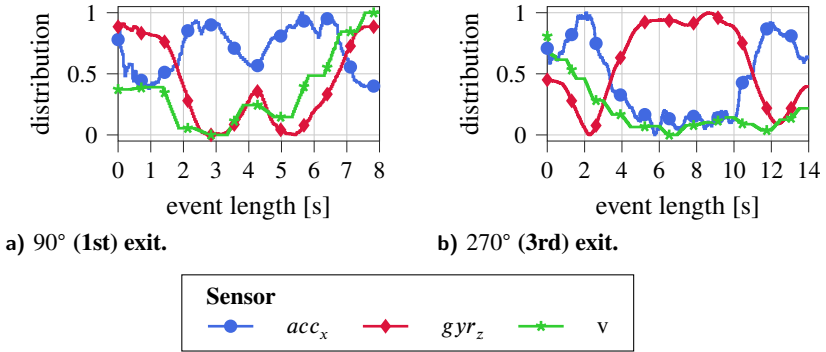
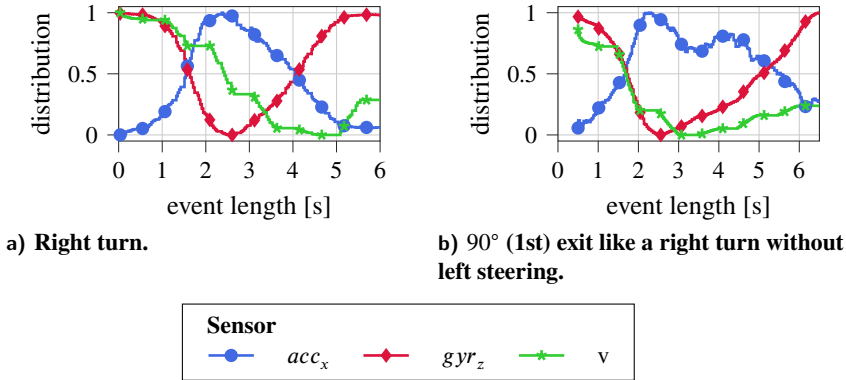


Figure 8.6 Patterns for traffic circle exits at 90° (1st) and 270° (3rd).

structure. One can see from Figure 8.6b, which shows passing and exiting the third exit, that the plateau is longer compared to the second exit as shown in Figure 8.5.

In-depth
analysis

The 90° exit pattern is not that unique compared to later exits: The turn to the left only lasts for a short period because the direction change is minimal in a traffic circle. Within gyr_z , a local maximum is apparent at 3 s in Figure 8.6a, but the apex does not deflect significantly higher than the other exits (c.f. Figures 8.5 and 8.6b). Compared to the other exits, leaving the traffic circle at the first exit at 90° does not require a substantial countersteer maneuver. It is even possible to depart a roundabout via this exit as if it were a normal right turn. This circumstance is shown in Figure 8.7. Shown is a plain right turn in Figure 8.7b. It is depictable that the gyr_z course decreases until it reaches zero and then returns to its initial value when finished turning. Also, acc_x shows a reduction in speed when beginning the turn with a resetting to the initial value afterward. The course is somewhat similar to a traffic circle pattern at the first exit (see Figure 8.6a), except for the absence of the peak in the gyr_z course due to a missing countersteer. Both cases can be differentiated. Next, Figure 8.7b also shows the passing of a traffic circle, but the pattern is more similar to a right turn and does not have a plateau. Consequently, driving style has a direct impact on the ability to detect patterns. As *ROADR* is a crowdsensing application, one can conclude that common traffic circle patterns are present in the data set. Hence, the objective is to detect distinct and unique patterns such as Figure 8.6a prior to Figure 8.7b.



Patterns for a right turn and a traffic circle passing without countersteer resulting in a pattern similar to a right turn. Figure 8.7

Traffic Lights

8.4.2

Traffic lights are light signs (according to §37 Road Traffic Regulations (Straßenverkehrs-Ordnung)) to control traffic flow at intersections and bottlenecks by alternating the clearance of incompatible traffic flows. Four light stages are defined, namely green as the release phase, yellow as the intermediate phase preceding the blocking phase, which is represented using a red light, and yellow-red as the precursor of the release phase. The phases are linked with a light schedule that corresponds to the opposite equivalent traffic light. The total circulation time is the sum of all four phases and depends on various parameters such as the speed limit or the distance. *RiLSA* [310] classifies circulation time into three categories based on duration with an expected circulation time of 90 s. Furthermore, *RiLSA* [310] also defines safety variables in traffic light circuits where no participants are allowed to drive.

Definition

The waiting time, i.e. red phase, is readable from FPD since the detection is based on the movement of the vehicle and the mobile device, respectively. Other phases are hard to detect since they may have no impact on the vehicle, particularly the release phase. Hence, we focus on the blocking time, which is a significant attribute in distinguishing the traffic light from other elements. We found that the expected blocking time at a traffic light is significantly higher and more reliable than stop signs [317]. In addition, traffic lights are often stationary at intersections that allow one to change directions. The FPD includes information on the direction of the turn after a traffic light to differentiate the light phase between different lanes.

Waiting time as the unique pattern

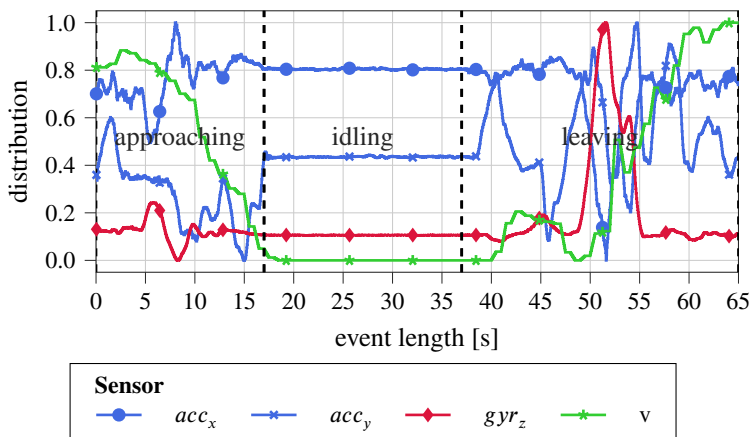


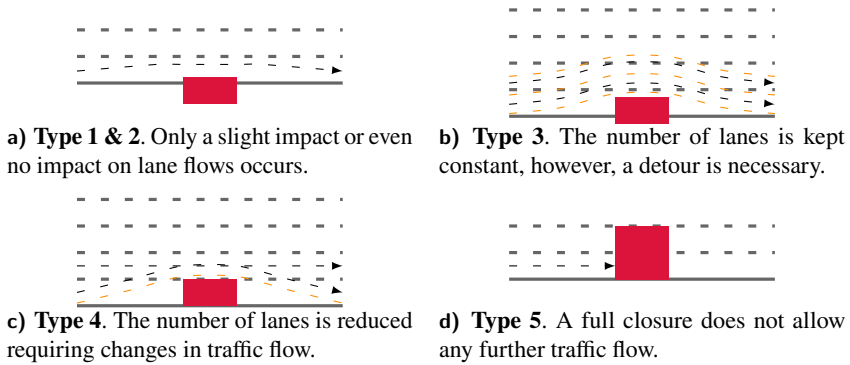
Figure 8.8 A traffic light passing has three different phases. These phases are approaching where on reduces the speed, idling while waiting for green light, and leaving.

Analysis of
traffic light
patterns

Figure 8.8 illustrates the sensor pattern for a traffic light. The vehicle *approaches* the traffic light and decelerates within a 10 s interval, which is also reflected in acc_y that is lower than in the following idle phase. The speed shows a short plateau before standing still because one is approaching the stopping line or the preceding car. The car comes to a complete stop while in the *idling* period. Although sensor readings for the accelerometer and gyroscope show minor variations, the vehicle is standing (c.f. Section 4.3.2). This noise is related to sensors and the vehicle's minor shacking (e.g. because of the engine). The *leaving* phase is dominated by acceleration maneuvers. The speed increases until it reaches a level comparable to that at the beginning of the approaching phase. This is because the speed limit is commonly similar before and after intersections. Multiple local maxima are present in the readings of the acc_y course during this phase which is related to the traffic situation, such as crossing pedestrians when turning. gyr_z explicitly indicates this turn. The velocity reduces before going up again; hence one cannot assume a reverse pattern to the approaching phase but a different one without linearity.

8.4.3 Road Work

Definition Certain types of road work are identifiable depending on their impairment of traffic. Of interest are road works where vehicles potentially have to change lanes, traffic jams can occur due to lane narrowing or similar. The German guidelines



Overview of different types of road works. Five different categories of road works have been identified, with only two kinds having a direct impact on a vehicle's trajectory and are therefore of interest. (based on Franz [S3] and Rsa [309]) Figure 8.9

for road works [309] serve as an orientation on the types that exist, based on which we conclude five categories of road works [S3]. These are presented in Figure 8.9. Only road works of types 3 and 4 (c.f. Figures 8.9b and 8.9c) directly impact a vehicle's trajectory and are therefore of interest within the *ROADR* platform. In addition, one can distinguish between road works in urban or rural areas. Highways post a different scenario, although their traffic flow pattern is similar to the one depicted in Figure 8.9b, except that they may have significantly long obstacles.

If a vehicle is driving in a lane where a road work is imminent, it must bypass the obstacle by changing lanes. This maneuver indicates *entering* of a road work and can be seen in Figures 8.10a and 8.10b in the readings of the gyr_z . This pattern shows a lane change together with a slow and ongoing reduction in speed as picked up by the acc_y . While *passing* the road work site, the vehicle maintains a consistent and lowered driving speed following the temporary and reduced speed restrictions seen at acc_y readings from 8 s to 20 s (c.f. Figure 8.10a) and from 22 s to 32 s (c.f. Figure 8.10b), respectively. When leaving the road work, the speed increases significantly as acc_y at 21 s in the first scenario. Also, in Figure 8.10b, the inversed gyr_z pattern compared to the entering phase shows a lane change back to the original lane before accelerating back to the incoming speed. However, a vehicle can remain in another lane as illustrated in Figure 8.9c where a vehicle chooses to remain in the upper lane as the recording of Figure 8.10a shows. Hence, the reversed gyr_z pattern is not to be expected as a strong requirement.

Impact on
movement
behavior

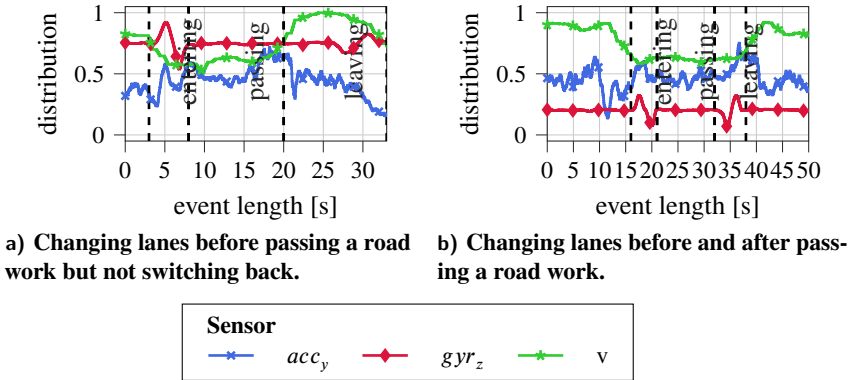


Figure 8.10 Overview of the phases of a road work passing. A road work has three different phases, namely approaching where on reduces the speed, passing the obstacle, and leaving.

Dependence
on the type
of road

One has to note that lane changes, as well as speed limits, may vary depending on the area. For example, lane changes on a highway may be much more extended than sharp ones in cities. Furthermore, a speed limit reduction from 130 km h^{-1} to 60 km h^{-1} in a road work is easier to identify and more unique than what we find in cities. Cities may only post a speed limit reduction from 50 km h^{-1} to 30 km h^{-1} , with the latter speed not uncommon during rush hours.

8.4.4 Road Quality

Structured
Literature
Review

The assessment of road quality can also be done with the help of sensor data. This also includes the detection of speed bumps and potholes. However, compared to the three events presented, traffic circle, traffic light, and road work, road quality is not defined by objective attributes but is subjective to drivers and passengers. Road quality, speed bumps, and potholes are of interest to academia for various reasons, including safety, durability, and maintainability. A SLR was conducted to survey the current status quo, the results of which will be presented here [S15]. Road quality papers sometimes include aspects of pothole detection and similar.

Threshold-
based and
ML-based
approaches

The methods to assess road quality can be classified into threshold-based and ML-based approaches. In total, 47 articles have been identified that match our criteria. The identified works are listed in Table 8.1 including their respective categorization and publication year. Although there are still new works dealing with threshold-based approaches, recent works mainly use ML-based methods to assess road quality. Interestingly, papers applying machine learning to the task also differentiate what results in lower road quality by including defects such

as potholes and speed bumps. Articles that do not extract some categorization (including simple good or bad bins) are not considered road quality papers. Most publications assume a fixed position of the recording device (i.e. smartphone) or mention such aspects. We are not focusing on the impact of vehicle and related problems (e.g. smartphone-to-vehicle alignment) here because that topic was already dealt with in Part I of this work. It should be noted that thresholds do not necessarily need to be static but can also be contextually adaptive [28]. Xue et al. [411] focuses on the practicability of road quality assessment because estimation is not only subjective to a driver, but the measurements captured by a smartphone in a vehicle depend on the dynamic properties of the vehicle itself.

Overview of the 48 publications identified in the SLR. Works are addressing specific aspects of the quality assessment with some only focusing of detecting road defects. Table 8.1

Publication	Year	Type ¹	RQ ²	PH ³	SB ⁴
Staniek [357]	2021	T	●		
Kyriakou et al. [224]	2021	M			●
Carlos et al. [64]	2021	M		●	●
Xue et al. [411]	2020	T	●		
Tiwari et al. [369]	2020	M	●		
Seid et al. [336]	2020	M	●	●	●
Kotha et al. [213]	2020	T		●	
Dimaunahan et al. [108]	2020	T	●		
Basavaraju et al. [37]	2020	M	●	●	
Badurowicz et al. [28]	2020	T	●		
El-Kady et al. [195]	2019	M		●	
Ali et al. [10]	2019	M	●		
Chuang et al. [84]	2019	T	●		
Dey et al. [105]	2019	M	●	●	●
Zhao et al. [425]	2019	M		●	
Kumara Thilakarathna et al. [222]	2019	T	●		
Wang et al. [392]	2019	M			●
Carlos et al. [65]	2018	M		●	●
Wang et al. [390]	2018	M		●	●
Souza et al. [354]	2018	M	●		
Sillberg et al. [342]	2018	T	●		
Li and Goldberg [230]	2018	T		●	●
Kataoka et al. [203]	2018	T	●		

continued on next page

Cabral et al. [59]	2018	M	●	●	●
Ameddah et al. [18]	2018	M	●		
Singh et al. [346]	2017	M		●	●
Silva et al. [343]	2017	M		●	●
Harikrishnan and Gopi [166]	2017	T		●	●
Alqudah and Sababha [15]	2017	M	●		
Allouch et al. [12]	2017	M	●		
Lima et al. [233]	2016	T	●		
Kalim et al. [197]	2016	M		●	●
Gawad et al. [139]	2016	M		●	●
Amirgaliyev et al. [19]	2016	T	●		
Sharma et al. [337]	2015	T		●	●
Yi et al. [416]	2015	T		●	●
Vittorio et al. [382]	2014	T		●	
Douangphachanh and Oneyama [115]	2014	T	●		
Douangphachanh and Oneyama [114]	2014	M	●		
Astarita et al. [25]	2014	T		●	
Douangphachanh and Oneyama [113]	2013	T	●		
Syed et al. [364]	2012	T	●		
Fazeen et al. [125]	2012	P	●	●	●
Astarita et al. [24]	2012	T	●		
Aksamit and Szmechta [7]	2011	T	●		
Mednis et al. [261]	2011	T		●	
Mohan et al. [268]	2008	T		●	●
Eriksson et al. [122]	2008	M	●	●	
<i>Sum</i>			28	24	18

¹ M (Machine Learning) / T (Threshold) ² Road Quality ³ Pothole ⁴ Speedbump/Hump

GPS as the
dominant
sensor

Most commonly, the combination of accelerometer and GPS is used to address the road quality [7, 18, 19, 24, 28, 113, 139, 195, 197, 222, 230, 233, 261, 336, 337, 342, 346, 354, 357, 369, 382, 392, 416, 425] with the GPS used to derive the velocity in most cases (apart from pinpointing the derived information to a location). Next, approaches include the gyroscope along with the accelerometer and GPS [10, 12, 37, 114, 203, 390]. A single approach used a combination of the accelerometer and the camera of the smartphone [213] which is questionable

from a privacy point of view but also from a practical perspective. Combining more sensors tends to improve the respective performance of a given approach, although this effect is not related to the threshold- or ML-based approaches.

There is no consensus, though, that different speeds have an impact on the accuracy of the detection (Zhao et al. [425] in contrast to Yi et al. [416]). Although sensor data changes with different speeds [19, 108, 425].

Dependency
on the
velocity

Various special orientations can be found in some works. For example, Zhao et al. [425] tries not only to detect potholes but also to measure them geometrically. Furthermore, road quality is also not always measured but e.g. attempts to distinguish between paved and unpaved roads [59]. Motivation for the work often correlates with the location in which the relevant studies have been conducted. India leads the way with six works [166, 203, 268, 337, 364, 369]. European countries are leading, except Italy [24, 25, 382], Finland [342], Cyprus [224], Poland [7, 28, 357], and Portugal [343] not being found. Comparability between works is infeasible because algorithms, especially threshold-based ones, are tailored toward the respective data set.

Special notes

Detection of Events

8.5

After introducing the mobile companion as a platform component to locally detect events in a privacy-friendly manner, we now explain how the detection modules for events, namely, traffic circles, traffic lights, and road work, are implemented.

We derive descriptive features as multi-dimensional properties of the time series. A feature is an individual measurable property of an observed process [71]. The properties of raw data are transformed into a more straightforward representation by extracting and converting information [123]. However, the constrained mobile environment is challenging. Common approaches to using hundreds of features, with many of them being irrelevant or redundant, are not feasible. Hence, unnecessary variables are to be eliminated to improve the performance of the ML model [71]. Feature engineering is the task of extracting and selecting features accordingly. It has three objectives: improving prediction performance, providing fast predictions, and producing cost-effective predictors [117]. Due to CPU and battery constraints, the mobile environment explicitly demands a streamlined workflow with fewer but significant features. At the same time, accurate prediction is unquestionably decisive. Thus, we perform a combination

Development
of descriptive
features for
events

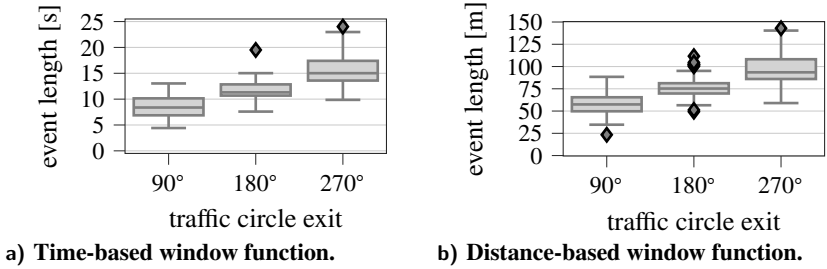


Figure 8.11 Relationship between parameters of the window functions and the respective traffic circle exit. Clusters are formed based on the exit taken, indicating a correlation between time and exit. There are no significant overlaps considering the clusters indicating that the sensor data can be used to identify the exit.

of automatic and manual feature engineering and selection to derive a minimal set of features addressing the said constraints and requirements.

8.5.1 Traffic Circle

Sliding window-based detection

The detection of traffic circles is based on sliding windows and depends on the exit taken, as confusion with curves is possible with earlier exits, as described. First, the corresponding windowing method was defined based on the test data. A robust windowing method is desirable. This means that clusters must form according to the exit of the traffic circle, so detection is possible. Desirably, there should be little or no overlap between the clusters to minimize confusion in exit prediction. Distance-based and time-based windows M_{Wd} and M_{Wt} , respectively, are available for the time series data. Experiments have shown that both methods are equally suitable and provide recurring results as depicted in Figure 8.11. Furthermore, we need to define a threshold for when a window is classified as containing a traffic circle since it is unlikely that a whole window from start to beginning reflects a traffic circle. We call this parameter θ_{TL} .

Overlapping window parameter definition

Therefore, we use time-based windows by creating windows with the sliding window function W in combination with M_{Wt} . We set $\omega = 20$ s to include the longest traffic circle passing found in our data set. Furthermore, we use an overlap of $o = 0.6$ and threshold $\theta_{TC} = 0.2$ which means that 20% of a window need to contain a traffic circle pattern. We optimized all hyperparameters using a `sklearn-grid-search` using a Random Forest classifier in the ML pipeline that combines different parameters to sets and then evaluates their performance accordingly. For more information, refer to our work specifically on traffic circles [319].

The final model will be executed on users' mobile phones to preserve privacy. However, the training and evaluation of the respective ML model are done on the platform-side in the CDP domain. For each window, the features were extracted using *TSHFRESH* [83]. This framework helps to easily derive a variety of features for time series to use in ML tasks. We then trained a sequential NN with the 150 most promising features that were selected programmatically using an initial sklearn [291] pipeline with the Random Forest classifier to analyze the data (c.f. [319]). More minor feature sets result in shorter runtimes and are therefore desirable for mobile environments to fulfill our given requirements. We ported the necessary features to Kotlin to use them in the *ROADR* mobile application.

Learning the NN-model

Eventually, the trained TensorFlow model is converted to a TensorFlow-lite model that can be run on Android devices with hardware acceleration. This guarantees energy efficiency, fast detection runtimes, and, as a native API, high compatibility across devices. To further optimize resources, we apply prefiltering to the incoming data stream using CEP: only patterns that show a right turn are considered to be processed by the NN. This check is swift and can be performed only by analyzing some statistical features for gyr_z for a window. Therefore, a window can be efficiently discarded if it does not contain a traffic circle. The final model uses 100 extracted features from the raw data within a window and determines the traffic circles.

Deploying the model

Traffic Light

8.5.2

Traffic light detection is built similarly to the traffic circle detection module. Also, the approach is similar. The statistical distribution of traffic light events is analyzed to decide which type of windowing function is a feasible choice. The challenge is to find a window function and corresponding parameters that cover the range of traffic lights as much as possible. Only then is detection with appropriate quality possible.

Selection of the approach

Figure 8.12a shows the waiting time if the time-based function M_{W_t} is to be used. It shows a range of up to 215 s. On the contrary, Figure 8.12b represents the distance-based view (using M_{W_d}). Again, values between 26 m and 957 m are obtained. If the window size ω is too large, closely spaced traffic lights will not be detected. However, if ω is chosen too small, a traffic light pattern cannot be fully detected. If the window is too small, the same traffic light may appear in several windows. Especially in cities, both versions of the window function depend on external conditions, such as road traffic or speed limits. Therefore, we evaluated multiple window sizes for M_{W_t} as well as M_{W_d} . In conclusion,

Derivation of relevant parameters

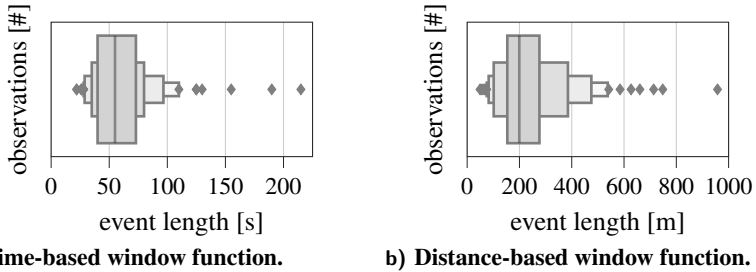


Figure 8.12 Relationship between parameters of the window functions and the respective traffic light event lengths. The distribution shows that both functions are suitable for the detection of traffic lights with slight advantages to the time-based variant

the distance-based version (M_{W_d}) performed superior to the time-based one. $\omega = 300$ m seems to be the best option within the data set. This seems logical, as this value is close to the median of the data (c.f. Figure 8.12a).

Discussion
on features

In addition, we define the threshold for elements in a window that must represent a traffic light pattern to be 60% ($\theta_{TL} = 0.6$). The overlap o is empirically set to 0.4. *tsfresh* was employed to calculate descriptive features for each window, which are subsequently refined during feature selection. Two different approaches were trained. One model tried to decide in one step whether it was a traffic light and, if so, in what direction a vehicle was moving in. However, this model was inferior to a two-step approach in which a ML model first decides if a window contains a traffic light before determining the driving direction. Feature engineering yields during the selection process a set of 10 features that are solely based on the y-axis of the accelerometer (acc_y) and the gyroscope's z-axis (gyr_z). GPS based measurements are entirely unneeded. An essential feature above all others is the idle time at the traffic light. gyr_z then helps to differentiate turn types based on the minimum and maximum derivations of that sensor after an idle time. Reducing the number of features to only 10 reduced the prediction runtime, not only during training and validation but during feature calculation in general by 60%. This will impact the detection within the *ROADR* application during the real-time detection of events.

Optimiza-
tions

A traffic light requires standing phases to be detectable. Hence, a fast check for standing phases was implemented to reduce the number of ML prediction activations.

Road Work

8.5.3

Road works are determined by analyzing specific turn patterns and comparing speed alterations with reference data from OSM. This detection module does not apply ML techniques but works using a state machine. First, a lane change must occur, which is detected using the gyroscope's z -axis (gyr_z). Recall that a lane change is a mirrored pattern with similar peaks in the positive and negative directions (c.f. Figure 8.10). Therefore, it can be differentiated from turns. A lane change can be sharp or smooth in terms of peak size and period. Intensity also helps in the selection of lane changes in relation to road works.

Rule-based approach

A lane change must occur in combination with a speed reduction. Then, the usual speed limit, queried from OSM's Overpass API, helps to estimate the likelihood of a road work entrance. Speed limits may be cached on the local device to reduce network footprint and handle no-network reception situations. A constant and lower speed than the reference speed for a given threshold indicates road work. If that speed increases later, road work is assumed. A subsequent lane change may improve the road work estimation, as a lane change is a unique road work pattern. The length of the phase with reduced speed depends on the overall speed limit. For example, highways have a higher speed limit, and road works are allowed to cover more distance.

Impact of velocity on patterns

A distance-based windowing method M_{W_d} was chosen for the road work detection module. Due to persistent traffic flow, standing times do not provide information about the road work site. Furthermore, the window length was empirically set to $\omega = 600$ m with an overlap $o = 0.6$ to include a lane change as well as a constant and reduced velocity phase.

Distance-based windows

Road Quality

8.5.4

The task of the road quality module is to assess the smoothness of the road. As can be seen from the analysis of related work, the comparability of the various approaches is possible only to a limited extent. This is mainly due to the application area for which the method was developed. Approaches that draw a diverse picture in India, for example, are not applicable in Germany due to generally good road conditions. Therefore, the following threshold-based approach serves only as a Proof-of-Concept and can be further refined or replaced in the future. The approach is based on the idea that recorded vibrations allow conclusions about the smoothness of the street [28]. The detection of potholes and speed bumps is implicit. The idea of *ROADR* is to improve navigation [317] by proposing an additional route, as shown by Souza et al. [354].

Vibration-induced assessment

Self-learned thresholds

Vehicles monitor their resting position to establish a sensor baseline where only vehicle vibrations are present without environmental impacts. Ultimately, there is a change in the sensor data from the smartphone that must be taken into account. Without context, it is impractical to use absolute numbers or deviations. To avoid this constraint, we use the moving standard deviation calculated by `std` for the gyroscope. Note that only gyr_x and gyr_z are evaluated.

Overview of parameters

Three parameters are important in assessing road quality. First, the standard deviation constructed from the readings in the standing phases indicates the rest position, denoted as a tuple $\sigma_r = (x, y)^T$. Second, the maximum of observed standard deviations for the x and y axes, denoted as $\sigma_m = (x, y)^T$. These two values deviations may change over time to reflect new observations and successively trend towards a global maximum. They indicate the range of the sensor to create an upper bound, while the standard deviation for standing phases represents the lower bound. The third parameter is the sensor standard deviation in-between this range that represents the road quality.

Calculation of the actual smoothness

Let $\widetilde{\mathcal{M}}'$ be a sequence of all measurements of a window. We define a helper method $v(m_a, m_b)$ that returns the absolute velocity for the transition from m_a to m_b . First, all measurements during standing and moving phases are collected and transformed into a list of gyroscope measurements:

$$\begin{aligned}\widetilde{\mathcal{M}}'_{moving} &\leftarrow \left[\widetilde{\mathcal{M}}'_{i,gyr} \mid i \in \mathbb{Z} \wedge 0 \leq i < |\widetilde{\mathcal{M}}'| \wedge v(m_i, m_{i+1}) \neq 0 \right] \\ \widetilde{\mathcal{M}}'_{standing} &\leftarrow \widetilde{\mathcal{M}}' \setminus \widetilde{\mathcal{M}}'_{moving}\end{aligned}$$

If $\widetilde{\mathcal{M}}'_{standing} \neq \emptyset$, the baseline σ_r can be updated. We use the median `med` to avoid outliers. A threshold θ_{rs} defines the range of values to be considered. $G_{s,x}$ and $G_{s,y}$, respectively, are initialized with \emptyset .

$$\begin{aligned}G_{s,x} &\leftarrow G_{s,x} \cup \left[g_x \mid g \in \widetilde{\mathcal{M}}'_{standing} \right] \\ G_{s,y} &\leftarrow G_{s,y} \cup \left[g_y \mid g \in \widetilde{\mathcal{M}}'_{standing} \right] \\ \theta'_{rs} &\leftarrow \min \left(\theta_{rs}, \lfloor |\widetilde{\mathcal{M}}'_{standing}| / 2 \rfloor \right) \\ \sigma_r &\leftarrow \left(\begin{array}{l} \text{std} \left(\{ g_j \in G_{s,x} \mid i = \text{idx}(\text{med}(G_{s,x})) \wedge j \in [i - \theta'_{rs}, i - \theta'_{rs} + 1, \dots, i + \theta'_{rs}] \} \right) \\ \text{std} \left(\{ g_j \in G_{s,y} \mid i = \text{idx}(\text{med}(G_{s,y})) \wedge j \in [i - \theta'_{rs}, i - \theta'_{rs} + 1, \dots, i + \theta'_{rs}] \} \right) \end{array} \right)\end{aligned}$$

The current measurements can be cleansed using the baseline to only reflect the deflection during movement related to the road surface.

$$G_x \leftarrow [g_x \mid g \in \widetilde{\mathcal{M}}'_{moving}], \quad G_y \leftarrow [g_y \mid g \in \widetilde{\mathcal{M}}'_{moving}]$$

$$\sigma' \leftarrow \left(\begin{array}{c} \text{std}(G_x) \\ \text{std}(G_y) \end{array} \right) - \sigma_r$$

Based on the cleansed standard deviation σ' , we can calculate the current road quality, denoted as \blacktriangle . It represents the quality assessed on a scale from $\{0, \dots, 1\}$ with 0 indicating a defective road surface. Similarly, the maximum observed standard deviation from the gyroscope σ_m can also be updated.

$$\sigma_m \leftarrow \begin{cases} \left(\begin{array}{c} \max(\sigma'_x, \sigma_{m,x}) \\ \max(\sigma'_y, \sigma_{m,y}) \end{array} \right) & \sigma'_x < \theta_m \wedge \sigma'_y < \theta_m \\ \sigma_m & \text{else} \end{cases}$$

$$\sigma_m \leftarrow \left(\begin{array}{c} \max(\sigma'_x, \sigma_{m,x}) \\ \max(\sigma'_y, \sigma_{m,y}) \end{array} \right)$$

$$\blacktriangle \leftarrow 1 - \frac{|\sigma'|}{|\sigma_m|}$$

θ_m is an arbitrary threshold to filter for outliers, eventually degrading the detection, set to 0.6. Consequently, one can further refine the determined road condition by including the current speed. σ_m may be initialized with $(0, 0)^\top$.

Refactoring ML Models to TensorFlow Models

8.5.5

Conception, improvement, and development of the respective models were carried out using *scikit-learn* [291], however, to implement them in the Android-based *ROADR* application, these *scikit-learn* models must be ported to some compatible artifact. For the final classification and prediction in the mobile environment, we chose to build a NN with *TensorFlow* [252]. Incidentally, we can take advantage of hardware acceleration and implement the models natively by porting all *scikit-learn* models to *TensorFlow-lite* models. *TensorFlow-lite* models are trained on centralized infrastructure, i.e. *ROADR*'s CDP and then efficiently distributed to clients without the need for retraining. Such models can

Note on development and deployment

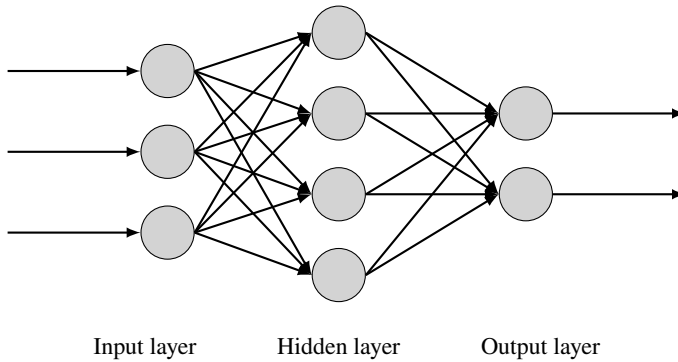


Figure 8.13 Example of a Neural Network with one hidden layer.

be executed on the mobile device using the official Android APIs. In each case, the traffic circle and traffic light models are a feedforward neural network with only one hidden layer, as shown in Figure 8.13.

Converted
feature
engineering

However, features as input for the models are derived using *TSHFRESH* [83] and self-engineered ones. They were ported from Python to Kotlin to be executable on Android natively. Only the required features are converted and unit tested to assert their performance and accuracy.

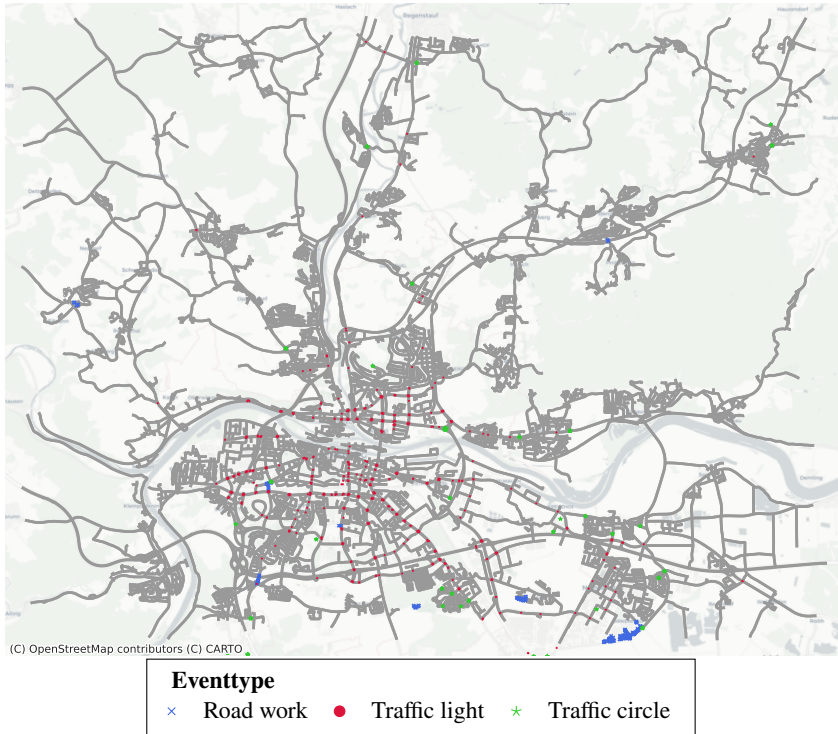
8.6 Evaluation

Setting The mobile companion and the respective detection modules are evaluated using arbitrarily chosen routes from the data set (see Chapter 5). Some routes may occur multiple times within the data set to account for differences in traffic patterns. We deliberately chose routes from the data set that were recorded using the *ROADR* application with a $f = 25$ Hz. Ultimately, our models were built and evaluated on 81 road works, 366 traffic lights, and 210 traffic circles collected in 1662 kms or 282 hs. An excerpt from Regensburg, Germany is shown in Figure 8.14

8.6.1 Preprocessing

Monitored
determina-
tion of
ground truth

To retain the ground truth of the events that occur during test drives, time frames were tagged with the type and time of the occurrence of the relevant event. This procedure was either semi-automated by extracting the necessary



Excerpt of traffic events at the time of evaluation for the respective area of Regensburg, Germany. In total 81 road works, 366 traffic lights, and 210 traffic circles are present in the data set. The total length of driven distance is 1662 kms covering different areas and traffic scenarios.

Figure 8.14

information from OSM (in the case of traffic signals or traffic circles) or was performed manually (in the case of road works). Humans have manually verified all labels. Events that occurred during the evaluation are excerptly shown in Figure 8.14. The time frames of related events have been designated so that each time frame captures the entire pattern of the corresponding events. The graphical representation of a single test drive was created using GPS data. The periods for the emerging traffic circles are chosen to begin immediately before entering the circle and finish soon after leaving. A few meters of straight-on driving were included to understand how a vehicle's orientation changes due to a traffic circle. Therefore, for each traffic circle passed during a test drive, the start time, the end time, and the chosen exit are recorded. Additionally, we

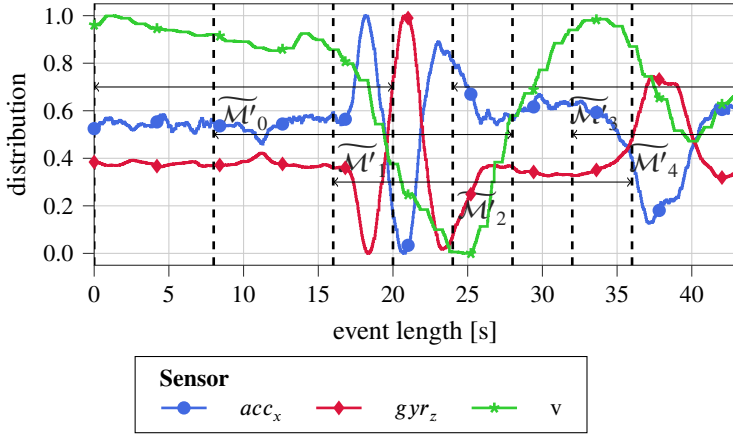


Figure 8.15 Pattern for a traffic circle that is divided into five sliding windows with a windows size ω of 20 s and an overlap o of 60%.

regarded traffic signals as such if they had a standing phase of at least 5 seconds. We captured the complete braking maneuver and acceleration behavior within the labeling process that occurs after or before a typical travel speed. Again, the event's period was recorded by each traffic light passed.

Avoidance of overfitting in ML models

Traffic circles and traffic lights are determined using Machine Learning. To minimize the probability of overfitting, we selected appropriate model adaptations and different parameters based on their performance on a training data set. k-fold cross-validation was performed to assess the performance of the various model parameters on the training data set. Finally, a second test data set was used to determine the generalization error of the correct associated model.

Note on sliding windows and ground truth

A critical aspect of the evaluation is modeling a real-time event detection situation. To replicate ROADR's capabilities, sliding windows are produced automatically for each test run using the window settings for the relevant detection module. An issue is the objective evaluation of the results. When sliding windows are constructed, an event may occur in several windows simultaneously. A traffic circle pattern is presented in several successive windows in Figure 8.15. In this example, two windows $\widetilde{\mathcal{M}}'_1$ and $\widetilde{\mathcal{M}}'_2$ entirely catch the event pattern, while nearby windows $\widetilde{\mathcal{M}}'_0$ and $\widetilde{\mathcal{M}}'_3$ gather only a portion of the event pattern. The model may be able to detect $\widetilde{\mathcal{M}}'_1$ and $\widetilde{\mathcal{M}}'_2$ and consequently may successfully identify the traffic circle event while failing to detect neighboring windows. Thus,

the ground truth is identified, but the model’s quality measure is erroneously degraded due to the model’s failure to recognize incomplete windows or duplicates. As a result, appropriate measures are applied in each case of event detection to quantify and assess the ground truth.

Accuracy

8.6.2

The following section presents and discusses the accuracy of the presented detection algorithms. For traffic circles and traffic lights, the performance of the final ML model is illustrated as implemented in the *ROADR* mobile companion application. The results of the intermediate models are omitted.

Traffic Circles

Performance of detecting a traffic circle including the correct exit. Shown are the results for the training and test set. Precision and Recall are rounded.

Table 8.2

	Training Data					Test Data				
	TP	FP	FN	Pr ¹	Re ²	TP	FP	FN	Pr ¹	Re ²
No Event	1534	35	1	97.8	99.9	749	9	3	98.8	99.6
90°	16	0	29	100.0	35.6	5	1	5	83.3	50.0
180°	59	1	2	98.3	96.7	16	1	2	94.1	88.9
270°	55	1	5	98.2	91.7	24	2	3	92.3	88.9

¹ Precision [%] ² Recall [%]

The classification of traffic circles is a four-class problem. Therefore, the algorithm must distinguish between no traffic circles and potential exits in the case of a traffic circle. As mentioned in the previous section, the algorithm evaluates sliding windows, and hence, successive windows may be detected as a traffic circle. However, we can safely assume that multiple neighboring windows consistently denote the same traffic circle considering the cautiously selected window length.

Multi-class problem

The results of the detection are shown in Table 8.2. It shows that the detection for the second and third exits is fairly reliable, taking into account that only two FPs are present. FNs are not critical, as the crowdsourcing environment allows the creation of conservative models in terms of recognition, but at the same time, high quality of the platform can be expected. We discuss that specific argument in Section 8.6.3. From this point of view, also the detection of the first 90° exits

Discussion of results

is also described as good since no FPs occurs. Although the recall appears low at 35.56 %, this effect was expected after analyzing driving patterns. Right turns are not trivially distinguishable from the first exit due to the dependence on the driving behavior. However, the high precision of 100 % proves the suitability of the model for the task. As shown, other driving maneuvers are hardly mixed with the unique traffic light pattern. This also includes right turns. For the test set, similar trends in performance are observed, indicating that the created model is generalizable and does not overfit.

Traffic Lights

Table 8.3 Performance of detecting traffic lights. Shown are the results for the training and test set. Precision and Recall are rounded.

	Training Data					Test Data				
	TP	FP	FN	Pr ¹	Re ²	TP	FP	FN	Pr ¹	Re ²
No Event	1580	111	7	93.4	99.6	350	33	6	91.4	98.3
Traffic Light	135	7	111	95.1	54.9	51	6	33	89.5	60.7

¹ Precision [%] ² Recall [%]

Discussion of
results

The challenge of detecting traffic lights is the variable period of still standing due to red lights or congestion. Therefore, such an event can be distributed across multiple windows, considering that *RiLSA* [310] defines up to 90 s for a traffic light phase. Through sliding windows, partial windows may result, in which the acceleration or deceleration phase is omitted. Additionally, these partial windows emphasize the standing-phase feature throughout the learning process, rendering it susceptible to misclassification as other types of traffic occurrences. Other incidents, such as pedestrians crossing the street, vehicles coming in, issues with abandonment, or stop-and-go traffic, also contribute to waiting times. Therefore, to avoid confusion, only windows that comprehend the complete pattern of a traffic light event are labeled accordingly, and such windows are used for training.

Unbalanced
data set

Subsequently, windows (partially) presenting a phase of acceleration and deceleration are analyzed in front of a traffic light. The detection performance is shown in Table 8.3. It is obvious that the data set is not evenly balanced, i.e. the number of traffic lights in contrast to the inverse class is low. However, that reflects a realistic scenario. The recall is mediocre for the training set, whereas precision shows strong values. This means that the detection algorithm often

overlooks traffic lights, but there is almost no confusion with other events. This is desirable in the crowdsourcing context. The test data set performs similarly. A slightly lower recall is a trade-off that can be considered since *ROADR* has many users reporting traffic events, resulting in a detailed general representation of reality.

Road Work

The road work detection algorithm was evaluated based on different types of roads. The measurements of the related event differ significantly depending on the shape of the road work. As explained, algorithms try to detect road works that induce an initial lane change with an optional lane change back to the originating lane. Hence, two patterns are of interest to this evaluation.

In general, 72 of 81 road works were detected successfully, resulting in a precision of 87.80 % and a recall of 88.89 %. The remaining nine road works could not be correctly categorized due to the failure to detect the initial phase of the road work in seven of these nine cases. The remaining approaches to road work do not include lane changes or accelerations towards the end of the road work. An acceleration phase that is missed may occur as a result of other traffic incidents, such as a red light after a road repair. Additionally, some roads may be devoid of lane changes, particularly rural roads and highways. Due to the greater variance in driving behavior, such as lane and speed changes, the identification algorithm is best suited for city streets, despite interfering traffic conditions: The algorithm shows an F1 score of 92.90 % in city traffic, but only 79.25 % on highways.

Discussion of results

Another ten patterns were detected as a road, even though they were other traffic events. The problem with the pattern was driving onto a highway whose traffic flowed only slowly. By pulling into a gap between vehicles, there is a clear lane change similar to a road work site. Similarly, false detections occurred only on roads with more than one lane. Optimizing the algorithm only to detect road works with a conclusive lane change improves detection by increasing precision. This ultimately removes eight out of ten wrongly detected road works but at the same time reduces the recall by around 5 %.

Problematic patterns

Road Quality

A supervised approach was used to assess road quality. The application was adapted accordingly by specifying the weights of the gyroscope axes to analyze road quality accurately. Several drivers used a qualitative scale to grade the road

Discussion of results



Figure 8.16 Road quality calculated by the *ROADR* application for all trajectories in the data set. Overlapping segments are averaged in terms of road quality.

surface. The weights were modified iteratively so that the z-axis of the gyroscope was regarded unnecessary, whereas the x-axis is crucial, as it accurately depicts the vehicle's movement, particularly in the presence of potholes. Figure 8.16 illustrates how road quality assessment may look with lighter colors that indicate a smoother experience, while dark colors reflect defective roads.

Limited
added value

However, measuring road quality in developed countries such as Germany, where the road network is mature, adds little value, as the roads were rated mainly as good or very good, with some outliers. In the context of the *ROADR* platform, this module can be replaced by a recognition of the road surface (e.g. cobblestone or concrete), allowing for the inclusion of this capability, which has been available in a limited fashion in OSM.

Aggregation

8.6.3

The *ROADR* platform is based on crowdsensed data. As already shown in this work, this data is subject to varying sensor quality in terms of accuracy and reliability. The detection performance of the *ROADR* application has to be taken into account, too, when one wants to use data from a crowd. Despite all causes of error, the *ROADR* platform can provide a holistic, informative, and accurate picture of the real world.

Figure 8.17 illustrates a traffic circle passing from four different drivers, vehicles, and smartphones, each of them using the *ROADR* mobile companion to gather sensor data. Each trajectory, including a map view of the structure, is shown in Figure 8.18. Recall that the *ROADR* ML models have been optimized for high precision instead of recall. Although sensor signals indicate a meaningful pattern of a traffic circle, the ML model fails to recognize the traffic circle in Figure 8.17d. However, the other three trajectories have been successfully analyzed in real-time with a detected traffic circle (c.f. Figures 8.17a to 8.17c). Taking a closer look at the driven path, one can see that all passes are of higher dimension, i.e. the drivers do not take the first exit. In fact, the missed pattern (blue) is a 270° exit, which tends to be detected very consistently. The sensor data shows a plateau that is neither flat nor constant but has a minor deflection at around 6 s to 8 s together with a velocity that decreases slightly after leaving the traffic circle before increasing sharply.

Balanced detection in favor of precision

This example indicates significant variances across multiple crowd participants. Using Everyone-as-a-Sensor, it is still possible to gather meaningful information without the solid requirement for detecting every event by each user. Additionally, the considerable variation within the presented patterns demonstrates the importance of adaptive approaches to deal with the varied accuracy of the sensors and driving tendencies. As a result, we used ML models trained on data collected from various drivers to detect potential violations that may be superior to threshold-based methods.

Everyone-as-a-Sensor

In addition, crowdsourcing also allows locating blurred events, such as road works whose location varies depending on driving style and accuracy of the GPS. Therefore, they are challenging to detect. However, after multiple passes of the same road work from different vehicles, a clear pattern becomes visible, as Figure 8.19 illustrates. The dashed line is the derived road work since it starts and ends with apparent lane changes.

Recurrent recognition to cope with uncertainty

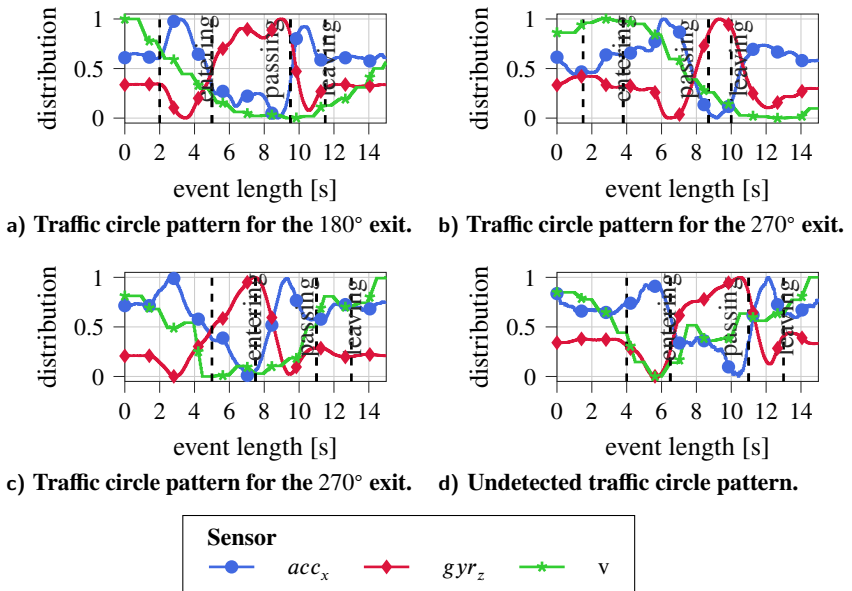


Figure 8.17 Different interpretations of the same traffic circle. The four different recordings of the same traffic circle by different drivers, smartphones, and vehicles show similar yet different patterns. Exits are of second or third dimension, although, only 75 % of all patterns have been recognized by the ML model.

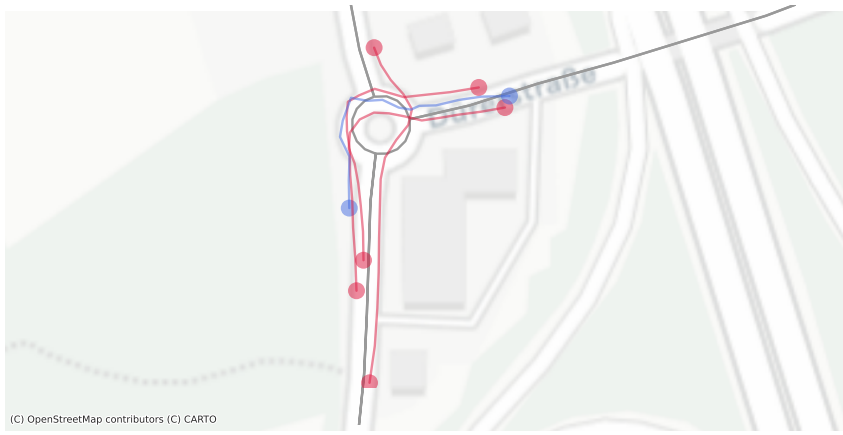
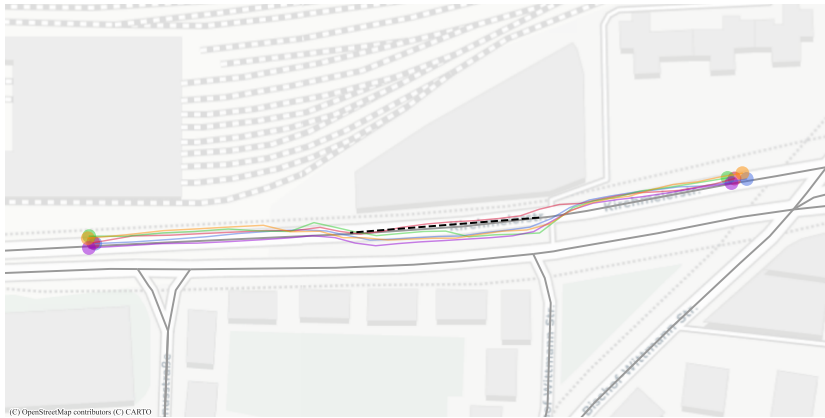


Figure 8.18 Trajectories of four different recording for the same traffic circle. Different exits were taken by multiple drivers. The blue path has been undetected by the ML model.



Five different passing of a single road work. It is illustrated that two lane changes allow to position a road work (dashed line).

Figure 8.19

Battery Consumption

8.6.4

A low-energy footprint is essential for user acceptance when circulating a mobile application. The *ROADR* application employs multiple energy-efficient methods, such as using hardware acceleration and reducing the number of computations. It was tested using three devices representing different classes of device ages.

- ▶ Google Pixel 4a (less than half a year old) with an estimated battery capacity of 3050 mA h
- ▶ Xiaomi Mi Mix 2 (about 2-3 years old) with an estimated battery capacity of 3000 mA h
- ▶ Samsung Galaxy S7 (about 4-5 years old) with an estimated battery capacity of 1800 mA h

The energy consumption while the application collects, analyzes, and eventually submits sensor data and events is shown in Figure 8.20. The ordinal axis indicates the percentage drop in battery life, while the abscissa axis indicates the duration of the application’s execution in minutes. After a 41-minute trip, the Samsung Galaxy S7’s charge level declined by 7%. The Xiaomi Mi Mix 2S’s state of charge reaches this point after 87 minutes, whereas the Google Pixel 4a reaches this point after 73 minutes. We ascribe the Galaxy S7’s considerably faster

Exemplary analysis

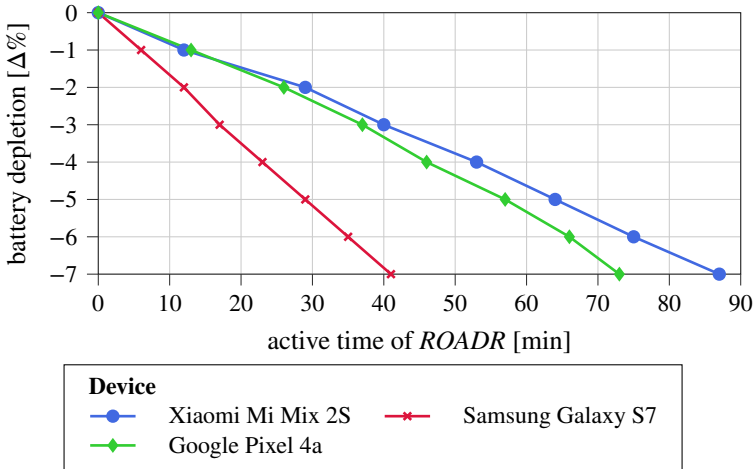


Figure 8.20 Overview of battery impact of ROADR on different devices. Shown is the time needed for 1 % of battery depletion while ROADR is active (Estimated battery capacity: Xiaomi Mi Mix 2S 3000 mA h, Samsung Galaxy S7 1800 mA h, Google Pixel 4a 3050 mA h)

discharge to two primary aspects. It features a slower CPU, which means that calculations take longer and require more energy. Furthermore, it is an older device that has been used in daily life and, as a result, has accumulated a large number of charging cycles, drastically reducing battery health [331]. This fact is further demonstrated by the AccuBattery application⁴, which calculates the maximum capacity of the battery, which was previously 3000 mA h, to be only 1800 mA h.

GPS as the
main energy
consumer

Altogether, the results demonstrate that the battery consumption of the proposed ROADR application is acceptable from a user perspective and hence feasible for everyday usage. Notably, the energy consumption can be significantly reduced by waiving GPS. Currently, GPS is gathered with a frequency of 1 Hz (the fastest interval available in Android) to either label data or preserve location information in case of a detected event. However, ROADR can be extended by requesting GPS on-demand for event detection to eliminate the need to query the location continuously.

⁴<https://play.google.com/store/apps/details?id=com.digibites.accubattery>

Conclusion and Outlook

8.7

With *ROADR*, we have presented an architecture that takes advantage of the potential of sensor data in road traffic. At the same time, the interests of the various stakeholders are protected by only transmitting aggregated, locally evaluated information when necessary, following the Privacy by Design paradigm.

A detailed examination of specific traffic events and structures reinforces the potential of crowdsensing. The chapter presents a retrofittable Android application that collects and analyzes sensor data in real-time. With the help of Floating Phone Data, traffic circles can be detected unambiguously and with high quality recurrently across different settings in some instances. A local Tensorflow-based Neural Network processes the data and, using a centrally learned model, can detect traffic circles in real-time and determine the exit. Using a second model, it is possible to identify traffic lights based on speed trajectories and idle times. An optimized rule-based matching algorithm uses steering and speed patterns to identify road works that significantly influence traffic events.

Events

In the sense of crowdsourcing, all this information is received and aggregated by the central, privacy-friendly *ROADR* platform. Respecting autonomy, users decide how to forward information. According to the data minimization approach (c.f. Section 6.1), the data received is deleted after evaluation. Furthermore, the crowdsourcing approach allows for a robust and balanced design of AI models and algorithms. This reduces the probability of incorrectly detected structures and ensures sufficient quality of map quality using a thresholding approach. *ROADR* allows crowdsensed information to be redistributed for the benefit of the public without introducing privacy risks. Also, the ML models can be extended and optimized based on the information received. Approaches of gamification [317] may increase the participation of users.

Platform

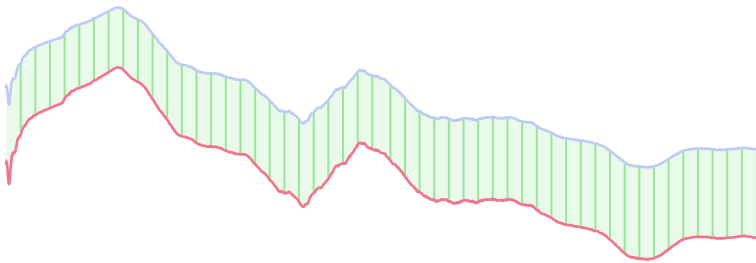
ROADR is capable of accurately estimating road quality. However, this assessment has been shown to be subjective and largely monotonic during the evaluation. This is due to the high quality of the German road network, which, compared to the related work presented, does not have any significant damage that must be imminently alerted. Here, a determination of the road surface instead of a subjective quality assessment is more appropriate and can be the subject of future work. Further improvements lie in the use of the GPS as an energy-inefficient sensor. An on-demand-based approach, similar to that addressed by Verma et al. [380], is conceivable and would eliminate the need for accessing the GPS for evaluation as all features are independent of the location. Not accessing the GPS may also increase the perceived privacy of users. *ROADR* requires GPS only for semantic

Outlook

enrichment of location information for detected events, but not continuously for recognition. Thus, if an event is detected, the location is required; however, this can be requested once the event is detected, and any discrepancies in location and event detection can be determined using interpolation. Chapters 4 and 12 show that the sensor information is sufficient for such an approach: Steering movements allow for detection of the trajectory, and the accelerometer can determine the speed.

Part III

Privacy Threats



Usage-Based Insurance: Pay-As-You-Drive and Pay-How-You-Drive

New insurance models in the vehicular environment promise usage-dependent rates that compensate clients for cautious or restrained driving. Usage-Based Insurances (UBIs) are in line with times and are now offered in many countries, including Germany, where data protection is of particular interest. Despite the lack of transparency of the models, they are enjoying increasing popularity; by 2020, 20 % of all vehicle insurance policies in Germany alone are expected to be of this type [378].

Figure 9.1 illustrates the reasons for traffic accidents in Germany from 2010 to 2020. One can see that speeding, the distance to other traffic participants, and turning are majorly responsible for accidents. Thus, an insurance company might have an interest in taking these factors into account when calculating the premium to enhance pricing accuracy and perform correct risk classification, eliminating the need for cross-subsidies. By including variable factors in the premium calculation, it might be possible to educate or incentivize people to drive more responsibly and, in countermove, offer reduced premiums. In fact, individuals are more willing to disclose information when they can prevent financial losses than when offered additional money [1, 111]. There are multiple benefits of Usage-Based Insurance (UBI) not only for the insurance company and the users, according to Husnjak et al. [184]. Firstly, there are **social** benefits like reduced accident frequency or less pollution and traffic congestion, as drivers are motivated to drive more foresighted (i.e. using an appropriate distance to the car in front and brake less (severe)). In addition, the reduced chance of accidents

Risk-based
insurance

Acknowledgement

Parts of the research presented in this chapter are based on work published previously [322].

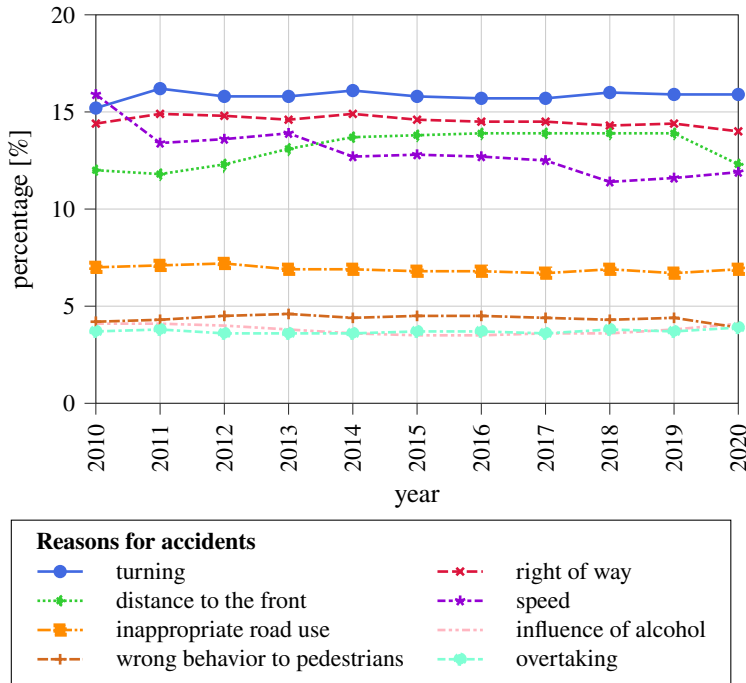
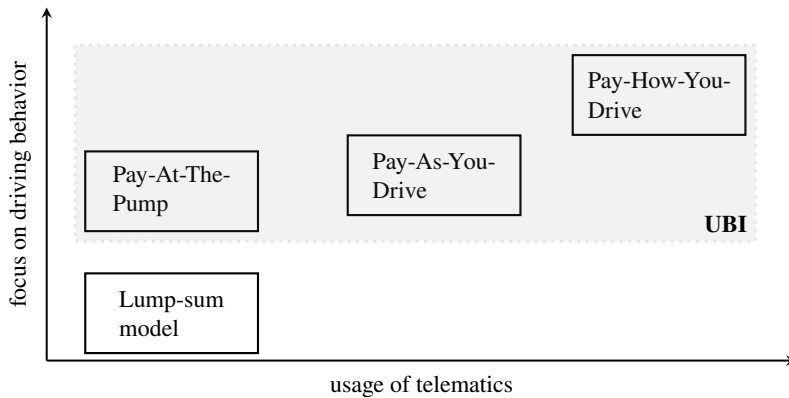


Figure 9.1 Distribution of reasons for accidents in Germany between 2010 and 2020. Most of the accidents occur when turning or for disregarding the right of way and distances. Speeding violations are on the rise as a cause of accidents. (data from Bundesamt [57])

and the enhanced efficiency of claims processing are beneficial for the **economy**. Lastly, the **environment** also benefits from drivers being more attentive due to reduced fuel consumption and, therefore, CO₂ output (owing to fewer braking and accelerating) or less road congestion. All benefits can be mapped to the reasons for accidents, as seen in Figure 9.1.

Data-driven
premiums

UBI can be considered a collective term for different types of insurance models that are based on telematics. Historically, the lump-sum payment model required customers to pay for their premium based on shared properties, such as their age, experience, and anticipated yearly mileage. One can argue that the risk of having an accident increases with the annual mileage, while, on the contrary, the experience increases. Nevertheless, there are obvious drawbacks in the lump-sum model that motivated more sophisticated approaches, each differing



Overview of variants of UBI. They can be categorized in the amount of telematics used and the behavior of a driver included in the assessment. (based on Husnjak et al. [184])

Figure 9.2

in the amount of telematics used and focusing on the behavior considered (c.f. Figure 9.2). In the following, we briefly describe the different types in chronological order based on Tselentis et al. [374]:

Pay-At-The-Pump (PATP) is an early version of mileage-based insurance. The idea behind this approach is based on the fact that the driven distance and the fuel consumption are somehow correlated. Here, a surcharge was considered to be paid while refueling the car. However, due to the different fuel efficiency between vehicles, the fundamental correlation claim is biased, and the Pay-At-The-Pump insurance scheme is not fair and has little benefit over standard lump-sum approaches.

Pay-As-You-Drive (PAYD) is a first approach to using telematics to calculate the premium by relying on e.g. GPS information to gain information about mileages, driven road types, and times. Thus, it relies on the user's behavior, such as his travel time. Pay-As-You-Drive (PAYD) is closely related and is sometimes used interchangeably with Pay-Per-Mileage (PPM). However, the difference lies in the data gathering method. While PAYD uses telematics, Pay-Per-Mileage is self-report-based insurance, where the premium is calculated based on the reported mileage of the driver [184].

Pay-How-You-Drive (PHYD) is a behavior-based approach to calculate the premium by enhancing PAYD to also use information from sensors

(either built into the vehicle or using additional devices) to derive information relevant for the calculation of the premium. Recalling the reasons for traffic accidents (see Figure 9.1), using sensor-based information allows e.g. to get information on sharp braking maneuvers or risky acceleration.

In addition, having telematics onboard offers a multitude of use cases such as tracking stolen vehicles via GPS, automatically making an emergency call in case of an accident, or reconstructing an accident.

Contribution While Husnjak et al. [184] analyzed UBI rates in 2015 worldwide, we focus on companies offering recent UBI premiums in Germany and derive a standard model for such insurance schemes. In particular, we provide

- ▶ an overview of current premiums offered in Germany that have PAYD and Pay-How-You-Drive (PHYD) features,
- ▶ an understanding of the UBI process and processed data with a derived meta-model, and
- ▶ a discussion on potential privacy threats originated from these models.

Structure Section 9.1 presents insurance companies that offer UBI premiums. Next, a meta-model is presented in Section 9.2 that explains the process of UBI. Subsequently, an in-depth analysis of the premiums is provided (see Section 9.3). This chapter concludes in Section 9.4 with an open question on privacy issues in this new field of insurance models.

9.1 Overview of UBI rates in Germany

Companies A field study was first carried out to obtain an impression of modern insurance models of the UBI form. The focus was on insurance companies in Germany, where data protection is historically strong. Concretely examined were tariffs of twelve companies, namely Aachen Münchner, Allianz, AXA, Cosmos Direkt, Friday, Generali, HDI, HUK-Coburg, Signal Iduna (sijox), Sparkassen Direktversicherung, VHV, and Württembergische. The list includes the largest insurance companies in Germany. With Friday, however, a young competitor (founded in 2017) was also considered, who, unlike other candidates, is mainly assigned to the motor vehicle insurance industry.

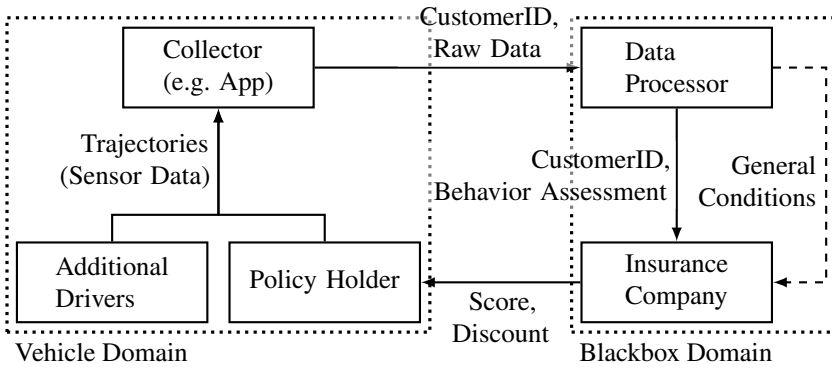


Figure 9.3

Derived meta-model of UBI after analyzing twelve premiums. Trajectories assigned to a vehicle are generated by either the policyholder himself or by additional drivers. They are processed by an external data processor that assess them within a blackbox process. The insurance company receives a behavior report that is eventually used to calculate the premium for that vehicle.

The insurances were analyzed with regard to various attributes such as general conditions (e.g. admitted participants), the information collected and the collection process, the information processing process, and other special features. Further aspects such as costs or discounts for customers as a diversification feature for the category of UBI compared to classic vehicle insurance were also considered.

Aspects considered

Meta-model of UBI schemes

9.2

Privacy has been identified as an essential criterion for insurance companies concerning customer communication. Consequently, the insurance process was adapted to these requirements. After the analysis of the twelve candidates, a meta-model could be identified, which is presented in Figure 9.3. The derived meta-model tends to be similar to approaches known from the literature (e.g. [184]). However, we can specify who processes and collects what information, which is crucial for privacy and, ultimately, user perception.

The General Data Protection Regulation defines three parties, all of which are reflected in the process.

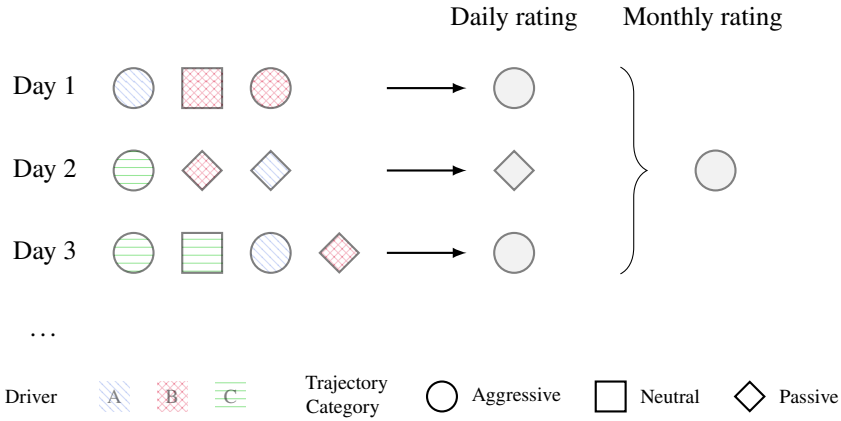
Stakeholders

- ▶ The **policyholder** is the person who has insured a vehicle with an insurer and, therefore, has entered into the contract with it. He pays the monthly installment for a vehicle. The vehicle can be driven by several additional drivers (up to 10, according to the study). In terms of GDPR, the vehicle domain is considered as the *data subject*.
- ▶ Furthermore, a *data controller* is present that defines the purpose and methods of processing personal data. As an **insurance company**, it describes to a user how the data is processed as part of the UBI product and how the monthly premium is calculated on it.
- ▶ Thirdly, there is a *data processor*, which handles data processing on behalf of the data controller. This is usually a third party. The data processor has the technical methods and knowledge, and it can describe the data collection process in terms of GDPR together with a data controller. This is called a *joint controller*. A data processor is likely involved in the collector design.

Procedure Within the process, a data subject generates the data while driving and submits the appropriate sensor data via a dedicated collector such as a smartphone application. Dedicated dongles are also conceivable (but they were mostly used them in the past, e.g. Sparkasse). They are connected directly to the vehicle's diagnostic interface, and sometimes have a companion mobile phone app (mixed form), but in some cases also have their interfaces. Within the black box domain, holding both the data controller and data processor, the user data is evaluated to eventually calculate the premium for a vehicle. Together with a customer ID, which identifies the vehicle, not the user, the data is transmitted to an external service provider, whereby several insurance companies generally use the same service provider. This service provider evaluates the data using non-communicated methods and extracts driving events, communicating them to the insurer. It is evident that the data processor determines the structure of the evaluation of the results since insurance companies that use the same processor transmit similar or identical information to the customers; hence, the black box may be considered a joint controller. Therefore, insurance companies are likely to outsource UBI technology. It has to be noted that, at least in Germany, the scoring criteria and the assessment process are a business secret and do not require publication¹.

Premium calculation Feedback on the trip is given to the driver quantitatively in the form of points (typically 0 to 100) or qualitatively using e.g. medals (gold, silver, bronze)

¹ German Federal Supreme Court (Bundesgerichtshof, BGH), 2014, VI ZR 156/13



Overview of PHYD process to derive a monthly premium. Data is collected for every trajectory made by different drivers to calculate the monthly discount for a single vehicle. The aggregated ratings are independent of the driver as they are assigned to the insured vehicle.

Figure 9.4

(Allianz)). Hence, trips of a day are aggregated independent of drivers to derive a vehicle’s daily driving class. Figure 9.4 illustrates the process of obtaining the monthly rating. Each shape indicates the driving style, i.e. aggressive, neutral, and passive, while the colors represent drivers. Let us say that on day one, there were three trips. Drivers A (orange) and B (blue) completed the trips with A being aggressive and B beginning a mixed form of aggressive and neutral. Using a trivial approach of averaging the three trips’ styles, the daily trip rating is aggressive. On a monthly scale, two aggressive days and one passive day yield an aggressive month. According to the meta-model, it is irrelevant which driver contributes to the monthly rating, although one can see that drivers e.g. driver C contributes significantly to aggressive trips. In some cases, minimum conditions are set for dynamic pricing, e.g. minimum distance per time unit (Generali) or some trips per time unit (AXA). An exception to this model is the VHV premium, where the evaluation is claimed to take place exclusively in a small hardware box, and only aggregated information is forwarded. However, unlike most other tariffs, additional costs arise per month for the VHV solution. Only Württembergische and CosmosDirekt consider negative consequences compared to a non-UBI premium, which conflicts with user perception [2].

9.3 Overview of premiums

Table 9.1 gives an overview of the premiums analyzed in the study, including the data processor, the collection method, and the features used to calculate the premium. We will now discuss each of them.

Table 9.1 Overview of the premiums analyzed in the survey offered by German insurance companies. Most approaches collect several items of information, mostly using a dedicated app. This information is then used to derive features in the context of PAYD and PHYD.

Premium	Processor	Data	Features								
			PAYD		PHYD						
		Hardware App	Time of Day	Road Type	Duration or Distance	Mobile Phone Usage	Acceleration	Braking	Cornering	Velocity	Raw
Aachen	MyDrive	●		●			●	●	●	●	
Allianz BonusDrive	IMS Inc	● ●	● ●				●	●	●	●	
AXA Drive	MyDrive	●					●	●	●	●	
CosmosDirect BetterDrive	MyDrive	●	●				●	●		●	
Friday	Friday	● [†]			●						
Generali	MyDrive	●	●				●	●	●	●	
HDI DiamondDrive	(un- known)	●	●	●			○	○	○	●	
HUK-Coburg Mein Auto	HDD GmbH	● ●					●	●	●	●	
Signal Iduna sijox AppDrive	Akquinet	●	● ●			● ●	● ●	● ●	● ●	● [‡]	
Sparkassen Direktver- sicherung	Telefon- ica	●	● ●				●	●			

continued on next page



Telematic dongle offered by insurance company HUK-COBURG. The dongle is installed on the front panel and communicates with a smartphone application. [I5].

Figure 9.5

VHV Telematik- Garant	Akquinet	●*	● ● ● ● ● ● ● ●
Württembergis- che	Vodafone Automa- tive	●	● ● ● ● ● ● ● ● ‡

* Applies local data processing † Submission of data via website ‡ Evaluated to detect speed violations

Data Acquisition

9.3.1

Data needed for UBI-based premiums can be gathered in multiple ways. In the past, it was common to use a dedicated device to be installed in the vehicle, which typically uses a car diagnostic port called OBD to collect the relevant data. Although this approach still exists, it is more common to use application-based approaches or dongles that sometimes require technical experience to be installed. While the application-based approach uses smartphone sensors such as the IMU or the GPS to collect data, standalone dongles have built-in sensors (i.e. not relying on OBD data). Dongles may be rented, bought, or given free to a customer. All in all, application-based premiums are mainly found in the wild.

For instance, HUK-Coburg uses a combination of dongle and application. The dongle (Figure 9.5) has to be installed on the front panel and collects data using built-in sensors. The application is used to forward those data to the insurance company. Some dongles can send data on their own, thanks to a built-in cellular modem.

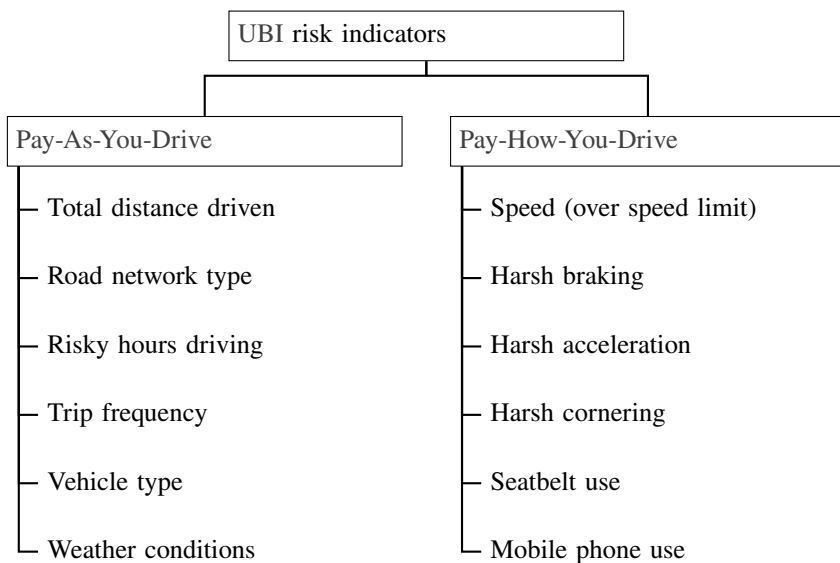


Figure 9.6 Overview of risk indicators in UBI insurance schemes. The two specifications PAYD and PHYD describe different aspects of a trip and a driver (based on Tselentis et al. [374])

9.3.2 Features

Tselentis et al. [374] presented different risk indicators as previously found in the literature, which are shown in Figure 9.6. Our study confirms that most of them are present and relevant in real-world insurance schemes offered by the twelve selected insurance companies in Germany. The analysis has shown that insurance companies usually use a mixed form of PAYD and PHYD in their UBI premiums (c.f. Figure 9.7a and Table 9.1).

PAYD PAYD-based premiums take into account the time of the trip to probably derive risky hours. The type of road is also commonly used since severe accidents in urban scenarios are more likely compared to rural areas according to IIHS data². In addition, insurance companies also gather the distance covered or the duration of a trip in some cases. Vehicle type and weather conditions are not collected using an onboard device since they can be determined by an insurance company using customer data or external services.

² <https://www.iihs.org/topics/fatality-statistics/detail/urban-rural-comparison>

In contrast, PHYD features are dominated by braking and acceleration events related to behavior and velocity (partly relative to the permitted speed) or as a general rule (CosmosDirekt: Speeds above 160 km h^{-1} are generally considered risky). In some cases, the speed is also set in relation to the permitted speed limit, so speed violations are also included in the evaluation. Seat belt usage was not collected; however, it is required by law to be buckled in Germany. Lastly, mobile phone usage is only used, although it is easily collectible using IMU data from a smartphone.

PHYD

Our findings indicate a development in contrast to Husnjak et al. [184] where mostly GPS-based metrics were used. Thus, recent developments include more sophisticated features to assess the driving style that was already mentioned in theory by literature [374]. However, driving behavior is used in lieu of travel behavior, such as vehicle maintenance conditions, which is in line with the literature [374]. A reason for this could be that vehicles in Germany are monitored by the Technischer Überwachungs Verein regularly; therefore, they have a basic safety and fitness to participate in road traffic.

Recent developments

Processing

9.3.3

During the analysis, six data processors could be identified. Figure 9.7b shows that they are not always located in Germany, even for German insurance companies. Consequently, data from German customers flow abroad. This aspect should be openly communicated to support transparency for and trust of each driver of a car. Data protectionists have therefore proposed that the collection of data in the vehicle should be indicated by a sticker, which the insurance company refused in the specific case [378]. However, which data is exchanged between data processor and insurer is not apparent and can be considered a black box. Interestingly, some companies are more specific in terms of the collected data and their processing (e.g. expired Sparkasse product), while others, such as VHV, even describe that they collected raw sensor data without specifying the derived maneuvers. In some cases, it is not evident what data are extracted.

Data evaluation abroad

The Formulated Privacy Problem

9.4

The insurance companies commonly apply pseudonymization techniques to convey a sense of security. Pseudonymization is an essential component to establish trust [207] in the UBI concept. It is evident that privacy is based exclusively on outsourcing data processing to an external service provider,

Example of communicating trust



a) Distribution of the applied methods in UBI products. It is common to combine features from PAYD such as time with more sophisticated and driver-related elements from PHYD like braking and accelerating.

b) Distribution of the data processor location. The processor performs the classification task on behalf of the insurance company. Only half of the insurance companies rely on Germany as the location for performing the actual processing of the raw sensor data.

Figure 9.7 Distribution of the applied methods of the UBI products and the origin of the data processor. The data processor classifies the data subject's driving style based on the raw data.

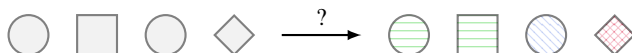


Figure 9.8 Process of the privacy invasion of UBI products. The upcoming privacy threat lies in the ability of an insurer or data processor to re-identify the originator of a trajectory. If that is possible, data misuse become a serious threat for drivers and users of UBI products.

who – not comprehensible to the customer – only passes on aggregated data to the insurer using a pseudonym (i.e. customer ID); the insurance companies also communicate this. For example, Allianz advertises the separation of "*your personal data from your driving data*" and that "*the external service provider of Allianz that processes the driving data [...] does not have any personal data from Allianz and therefore does not know who the driver or the policyholder is*"³. However, sometimes risky events are presented to the user by the insurance company, making it necessary to have specific data. The literature has shown that a related privacy policy does not help to increase understanding of the data handling process.

³Quote translated

In the interest of data economy and expediency, only data that actively allows pricing within the framework of a UBI premium should be collected by the insurer. It should be viewed critically by the client in the sense of privacy if further information can be read out of the data allowing to increase knowledge. Furthermore, it may also decrease the fairness experienced by users and limit their acceptance [243]. Consequently, the question arises whether an insurer is in a position, based on the data collected, to conclude who has made the respective trip per day, ultimately resulting in a misuse of data (c.f. Figure 9.8). Similar questions have been answered in the insurance context w.r.t. human activities [194].

Imminent
threat
through
evaluation

Since UBI premiums only cover one vehicle, not the individual, a statement about the driver is not necessary. However, this information can be used, for example, to find out whether the vehicle is (illicitly) shared. This seems particularly interesting in car rental scenario, where additional drivers are a chargeable option. The misuse of the data seems to be lucrative. For example, when knowing the driver, an insurance company can derive daily routines and points of interest.

Misunder-
standing of
focus

Let us discuss (GDPR) Article 5, highlighting the critical parts of the business model. We assume at this point that it is clarified that the data is personal data (i.e. personally identifiable information), so that (GDPR) Article 5 applies. Some discussion of this may be conducted later. This is not meant to be a legal evaluation, but the author only wants to support the critical discussion of the topic. Clear answers are not intended to be given here, but discrepancies should be noted.

Discussion in
accordance to
GDPR

GDPR Article 5 (1 (a)) *[Personal data shall be:] processed lawfully, fairly and in a transparent manner in relation to the data subject ('lawfulness, fairness and transparency');*

Apart from lawful processing in the sense of being compliant with the law, the latter two requirements are of fundamental interest in the context of this work. Fairness implies that the evaluation of the data is done according to the expectations of the individuals. As an example in UBI, this could refer to the fact that, on the one hand, a driver is aware that his trip is recorded and evaluated for UBI to enable pricing. On the other hand, a driver can anticipate that if he drives objectively prudently, a higher bonus is expected and eventually granted. This does not mean that a trip can be negatively evaluated if a risky driving style is present. The GDPR also talks about transparency so that participants and the data processing process should be clearly defined. The study shows that the first point is met, but doubts are warranted about the second. GDPR Article 12 (1) requires that the information be presented in "*concise, transparent, intelligible*

and easily accessible form, using clear and plain language". However, the different backgrounds, attitudes, and expectations of the subjects that are crucial to the information disclosure process are not taken into account (c.f. Section 1.2), nor is a level of detail or the like defined.

GDPR Article 5 (1 (b)) *[Personal data shall be:] collected for specified, explicit and legitimate purposes and not further processed in a manner that is incompatible with those purposes; further processing for archiving purposes in the public interest, scientific or historical research purposes or statistical purposes shall, in accordance with Article 89(1), not be considered to be incompatible with the initial purposes ('purpose limitation');*

A provider for UBI collects the data for the purpose of evaluating a ride, which should usually be known to the parties involved in the process. However, it should be noted that technically there is no way for the driver to enforce this accordingly. This aspect becomes interesting from the point of view of the evolving ML models. The unknown future uses are often not assessable [226], which also has an impact on the corresponding limitation. Again, technical certainties are desirable for a user. Even though a data processor has to explain the purpose to its users, he is biased. If the purpose of the data collection (i.e. the business model) is described too narrowly and precisely, this will ultimately condition a division of funds for change of purpose under GDPR Article 6 (4) in the case of minor changes. [39]. Such circumstances may be considered when analyzing the explanation of a data processor.

GDPR Article 5 (1 (c)) *[Personal data shall be:] adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed ('data minimization');*

The principle of *data minimization* (c.f. Section 6.1) also applies so that only data that is indispensably necessary for the evaluation should be collected. If raw data is transferred, as was seen within the study, this is to be regarded as questionable from this point of view. However, due to the covert process protected by trade secrets, it is unclear what data is necessary. To anticipate the findings of Section 11.4.2, it shows that identification (which is, therefore, the more challenging compared to classification) is quite possible with less data than the data collected by vendors. Consequently, the amount of data may be at the providers' discretion.

GDPR Article 5 (1 (d)) *[Personal data shall be:] accurate and, where necessary, kept up to date; every reasonable step must be taken to ensure that personal data that are inaccurate, having regard to the purposes for which they are processed, are erased or rectified without delay ('accuracy');*

As a rule, data should be up-to-date to create an evaluation that corresponds to a

user's current driving profile. This aspect is in the interest of both parties. This may also be relevant for the infrastructure, including the gathering device, such as the insurer's application on the user device.

GDPR Article 5 (1 (e)) *[Personal data shall be:] kept in a form which permits identification of data subjects for no longer than is necessary for the purposes for which the personal data are processed; personal data may be stored for longer periods insofar as the personal data will be processed solely for archiving purposes in the public interest, scientific or historical research purposes or statistical purposes in accordance with Article 89(1) subject to implementation of the appropriate technical and organizational measures required by this Regulation in order to safeguard the rights and freedoms of the data subject ('storage limitation');*

From a user's point of view, it should be sufficient to store the data for the assessment duration. Afterward, for example, it may be meaningful to collect only aggregated values, as shown in the example of this work. The technical and organizational options for deleting the data should also have a (technical) verification function so that the user can check this.

GDPR Article 5 (1 (f)) *[Personal data shall be:] processed in a manner that ensures appropriate security of the personal data, including protection against unauthorized or unlawful processing and against accidental loss, destruction or damage, using appropriate technical or organizational measures ('integrity and confidentiality').*

We can argue that a data processor has an interest in integer data as it is the foundation of his business model. Keeping the data confidential is an interest of the data processor for various reasons, including fines or loss of reputation. Hence, we may assume that both stakeholder interests are in line considering the sole data.

Now that we have found a potential privacy threat based on this new use of highly individual and user-focused sensor data, we will further investigate its extent. Therefore, this part is dedicated to side-channel attacks solely using zero-permission sensor data from a smartphone in a mobile environment. In addition to a comprehensive overview of different types of side-channel attacks, we also present two newly designed attacks that stress and demonstrate the sensitivity of the data in less constrained environments, such as the UBI scenario.

There is no doubt that sensors in smartphones enable a wide range of applications that can improve comfort, efficiency, or security, among others. For example, it makes perfect sense to align the screen of a smartphone with the help of accelerometer data or to make restaurant recommendations based on the location collected via GNSS. If the sensors or their data are used within the scope of the application, this can be termed a purpose-specific evaluation. If, in contrast, the data is misused and not evaluated for the purpose assumed by the user, this can pose a severe risk. In addition, unintended information leaks can also occur if an attacker uses information unrelated to a use case, e.g. the execution time of a specific operation [356].

The unintended usage of data is referred to as a *side-channel attack*. Here an attacker may use information gained from the system in a somehow non-intrusive way (e.g. via official APIs). An attacker then exploits the information to recover some leakage of secrets. In general, two types of side-channel attacks can be differentiated based on the origin of the data that is exploited for the attack. Data can be used unintendedly, or an attack can be carried out based on data published on purpose. Figure 10.1 illustrates the two classes with “unintended information leaks” being considered the traditional type of side-channel attacks [356]. However, this work focuses on the second type, namely “information published on purpose”, as this class may be viewed as the consequence of ubiquitous computing and modern data-driven business models such as UBI. Therefore, this chapter presents a survey on side-channel attacks with a focus on smartphone sensor data.

Side-channel
attack
definition

Side-channel attacks are favorable because they are not based on weaknesses in an implementation (e.g. cryptographic algorithm) where no leakage might

Bonded and
continuous
attacks

Acknowledgement

Parts of the research presented in this chapter are based on supervised work [S10].

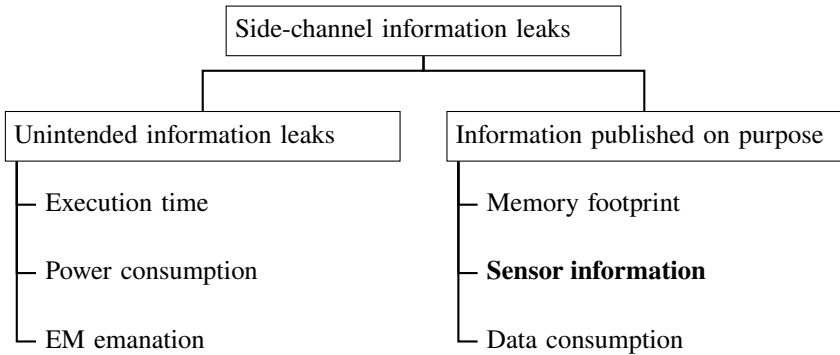


Figure 10.1 Overview of side-channel attacks in the sensor-based setting. Two types of attacks can be differentiated based on the origin of the data that is exploited for the attack. Data can either be used in an unintended fashion or an attack can be carried out based on data published on purpose (based on Spreitzer et al. [356]).

exist but are often inexpensive or feasible with simple equipment. However, they often require technical knowledge of the internal operations of the system. One distinguishes between bounded side-channel attacks, where the amount of leakage is bounded, and continuous side-channel attacks that allow the amount of leakage to increase continuously [303]. Consequently, side-channel attacks often result in an *information leak*.

Contribution This chapter supports the understanding of side-channel attacks based on sensor data generated by smartphones. We contribute with

- ▶ an overview of existing side-channel attacks in the given context and answers what specific goals are pursued,
- ▶ a proposal of a multiclass model to differentiate attacks based on the information extracted,
- ▶ a depth understanding of the structure of the respective attacks, and
- ▶ a discussion on protective measures according to the attack class.

Structure We first present the survey methodology, including research questions and a presentation of the document corpus in Section 10.1. Section 10.2 discusses related works also presenting to categorize side-channels attacks. Finally, in the next Section 10.3, we present a methodology to categorize sensor-based

attacks w.r.t. sensors used and information derived. Based on the methodology, an outlook of protective measures against the identified attack classes concludes this chapter with Section 10.4.

Structured Literature Review

10.1

We now present the methodology of our SLR that includes the research questions, search terms, and an overview of our document corpus.

Research Questions

10.1.1

The subject of the SLR is to outline “*the shape of side-channel attacks for smartphones in the literature*”. This includes an overview of what attacks exist in the first place. Furthermore, it will be clarified based on which characteristics the attacks can be distinguished in each case. Hence, the SLR should elaborate on what the ultimate goal of each attack is. Based on the derived characteristics, enablers of the respective attacks will be categorized in a newly developed methodology.

Thus, we define the following Research Questions (RQs) to answer the question.

- Q1** What side-channel attacks targeting smartphones are present in the current literature, and what do they target?
- Q2** What distinguishing characteristics exist in the separate attacks, and what are specific peculiarities?

Search Process

10.1.2

The search process for the SLR is based on Kitchenham and Charters [209].

The search focuses on ACM Digital Library, IEEE Xplore, ScienceDirect, and SpringerLink as a resource for publications. To grasp the relevant work, the following search string¹ was used as a basis:

```
SIDE CHANNEL
& ( SENSOR BASED | SENSOR* | SENSOR DATA )
& SMARTPHONE*
```

¹For an explanation of the notation, see Appendix B

```
& ( ATTACK* | LEAKAGE | EXPLOIT )  
& ~( SMARTWATCH* | WEARABLE* | IOT )
```

Smartwatches or other Internet of Things devices were deliberately excluded from the search radius, as it is expected that attacks differ due to the underlying characteristics (e.g. processing power, battery power, . . . and are not comparable). Furthermore, side-channel attacks should target recent OS with implemented permission systems. Hence, we include work from 2015 (the release of Android 6) to 2020. Furthermore, next to this forward search, a backward search was also employed to identify relevant work in the given research field. These yields work that is either often cited or particularly noticeable. It does not need to match the constraints for the forward search. Duplicates are eliminated.

The following inclusion and exclusion criteria are defined to assess the relevance of the articles. Infeasible work was then removed from the search corpus, while all other papers were analyzed in-depth with a full-text read.

1. First, attacks had to target the smartphone environment within a common setting specifically.
2. Then, only work was included that presents a novel approach for a side-channel attack.
3. This excludes all work that presents an overview of attacks (i.e. survey papers).
4. The attack had to use sensors from a smartphone device, which ultimately excludes all publications that use external setups such as external microphones.
5. Also, we eliminated work whose research methods were unclear or lacked scientific accuracy.

In addition, only peer-reviewed work in English was considered that was available to the authors. The peer-review process is assumed to provide only high-quality content with comprehensible attacks, although the quality was again assessed during SLR.

10.1.3 Relevant Findings

In total, 61 publications match the search criteria given. They are listed in Table 10.1. Additionally, the table categorizes attacks according to attack properties. Interestingly, except for eight attacks, most of the attacks are passive and do not

require any interaction with a user. This is even more severe than active attacks since the success rate can be increased in this way. The attack vector describes the way the attack is performed in the user domain. In general, the smartphone context allows two different directions. On the one hand, the attack can be executed using a dedicated application or hidden in a diversion application (as seen in Section 3.3). On the other hand, a web-based approach can be followed that does require a victim to open a bogus website. Most of the relevant works present attacks that are realized using a smartphone application, although some support both vectors.

Overview of the 61 publications identified in the SLR. Attacks are active or passive of a combination of both classes. Most attacks are Application-based. Table 10.1

Publication	Year	Passive Attack	Attack Vector ¹
Griswold-Steiner et al. [152]	2021	●	●
Zheng and Hu [426]	2020	●	●
Schmitt and Voigt-Antons [334]	2020	●	▲
Javed et al. [192]	2020	●	●
Cheng et al. [79]	2020		●
Ba et al. [27]	2020	●	●
Zhang et al. [423]	2019	●	▲
Perez et al. [293]	2019	●	●
Matyunin et al. [257]	2019		▲
Matyunin et al. [259]	2019	●	▲
Hodges and Buckley [173]	2019	●	●
Genkin et al. [141]	2019	●	●
Aliyu and Rahulamathavan [11]	2019	●	●
Zhou et al. [428]	2018		●
Wang et al. [394]	2018	●	●
Tang et al. [366]	2018	●	▲
Song et al. [353]	2018	●	▲
Ning et al. [280]	2018	●	▲
Murali and Appaiah [272]	2018	●	●
Matyunin et al. [258]	2018		○
Li et al. [231]	2018	●	●
Das et al. [95]	2018	●	○
Block and Noubir [47]	2018	●	●

continued on next page

Berend et al. [41]	2018	●	▲
Anand and Saxena [21]	2018	●	●
Song et al. [352]	2017	●	○
Narain et al. [277]	2017	●	●
Hua et al. [181]	2017	●	●
Davarci et al. [96]	2017	●	●
Chakraborty et al. [67]	2017	●	●
Anand and Saxena [20]	2017	●	●
Narain et al. [276]	2016	●	●
Mehrnezhad et al. [262]	2016	●	○
Kim et al. [208]	2016	●	●
Gupta et al. [159]	2016	●	●
Das et al. [94]	2016	●	○
Bartolini et al. [36]	2016		●
Shen et al. [339]	2015	●	●
Shen et al. [338]	2015	●	●
Ping et al. [299]	2015	●	●
Ho et al. [172]	2015	●	●
Biedermann et al. [45]	2015	●	●
Anand et al. [22]	2015	●	▲
Zhou et al. [429]	2014		●
Tobias Fiebig et al. [371]	2014	●	●
Spreitzer [355]	2014	●	●
Narain et al. [275]	2014	●	●
Yan Michalevsky et al. [412]	2014	●	▲
Genkin et al. [140]	2014		●
Dey et al. [106]	2014	●	●
Das et al. [93]	2014		●
Simon and Anderson [345]	2013	●	●
Xu et al. [409]	2012	●	●
Owusu et al. [288]	2012	●	●
Miluzzo et al. [265]	2012	●	▲
Han et al. [163]	2012	●	●
Cai and Chen [60]	2012	●	▲
Aviv et al. [26]	2012	●	●
Roman Schlegel et al. [313]	2011	●	●

continued on next page

Marquardt et al. [251]	2011	●	●
Liang Cai and Hao Chen [232]	2011	●	●

¹ ○ Web-based, ● Application-Based, ▲ Both

Based on the literature, a structure was implemented that allows comparing different kinds of attacks and generating additional features such as feasibility or scalability [209].

Related Methodologies

10.2

Classification models are developed for an intended purpose and are usually related to a specific issue. However, they have in common that they are intended to provide a quick, tangible, and structured overview of a topic [186]. The objective of the SLRs is to find correlations between sensors and attacks and to outline which pathways serve as enablers for information outflows.

The model of Spreitzer et al. [356] with a three-level classification is not suitable because it differentiates between active and passive attacks as the highest level of subdivision. Attacks are divided in the next step into physical and logical attacks. Here a distinction is made regarding whether the attack reaches its target by exploiting the corresponding hardware or software systems. In the third hierarchy level, attacks are divided into three categories based on attack proximity (near, surrounding, far). Attacks with different targets and attack paths are subsequently found as artifacts in these categories. This includes those, as mentioned earlier, sensor-based side-channel attacks. They are not different attack types but can be found in the different attack types. The three levels may not be suitable for classifying sensor-based side-channel attacks on smartphones due to the information derived from the SLR. This is in part due to the fact that almost exclusively passive attacks (53 to 8 active) are found. The model according to Spreitzer et al. also includes attacks that are not based on sensor data, such as, for example, the attacks mentioned at the beginning that exploit system properties such as the performance times of an operation or the energy consumption of a calculation. Furthermore, with one exception (“*Electromagnetic Analysis Attacks*”), all sensor-based attacks are lumped into the category of passive attacks with remote attackers. This lumping makes it impossible to achieve the intended overview. Nevertheless, a grouping of different attacks with a similar goal is adapted in the categorization model to be developed. No delineation is made based on the proximity of an attacker since sensor-based side-channel attacks do

Three-level model by Spreitzer et al.

not fundamentally require the proximity of the attacker; after all, they operate application- or web-based (c.f. Table 10.1).

Hypertext
Transfer
Markup
(HTML)5-
focused
models by
Diamantaris
et al.,
Marcantoni
et al.

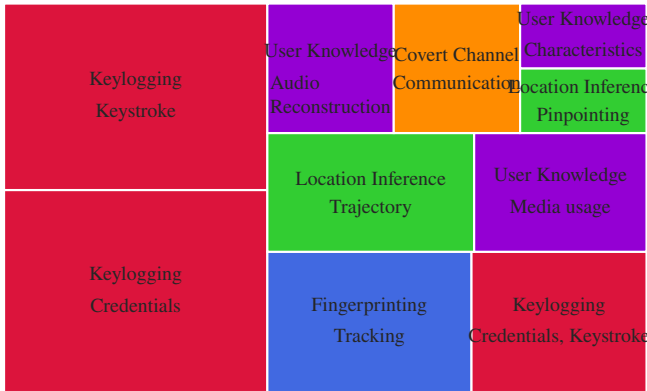
Another taxonomy focuses on mobile sensors in the context of HTML5 WebAPI and respective attacks [107, 246]. The authors take a different approach by introducing a classification of up to three levels, with the first level being attack classes, namely, “*physical activity inference*”, “*acoustic attacks*”, “*digital activity inference*”, and “*user tracking*”. As they claim, they purposely do not include any information about the sensors involved since multiple combinations of sensors can be used in the same attack category. They underline their statement by presenting recent attacks exploiting different types of sensors. Our document corpus suggests similarity, yet the sensors and attacks must be broken down to understand the structure of the attacks. Only then can particularly critical contexts be identified and appropriate countermeasures subsequently designed. Especially in the context of smartphones, the line between purposeful sensor use and unintentional information leakage must be examined in detail. A taxonomy should support downstream handling and protection accordingly, for example, through a PET.

Multi-level
model by
Hussain
et al.

Another taxonomy limits the subject area of keylogger attacks to one [185]. Furthermore, a division is made into smartphone-based and computer-based attacks on the first level. In the first category, sensor-based attacks can also be found with the same portfolio of sensors that we have also identified (see Section 10.3.1). The taxonomy distinguishes between IMUs, multimedia, and environmental sensors.

Meta-study
by Igere and
Williams

Igere and Williams [186] do not present a taxonomy but review a multitude of taxonomies for attacks and vulnerabilities in computer systems. Consequently, they aim to present common characteristics and requirements for the creation of a taxonomy, which is the motivation of our approach. Seven guidelines are defined, which will be briefly presented here. The guidelines focus on vulnerabilities and should support successive security evaluations. They can be broadly considered as a framework, even though attacks are not the main subject. First, a taxonomy has to be application- or system-specific to keep the subject tangible. Next, a hierarchical or layered approach is recommended with increasing granularity. Thus, we can start with the security properties like the known protection goals. Here, it can be seen that the identified attacks based on side channels are exclusively aimed at privacy/confidentiality. Thus, this distinction is obsolete. Furthermore, attack methods that become more precise with increasing hierarchy levels can be listed. It should be emphasized that classes must not be mutually exclusive, i.e. overlapping of the individual paths is conceivable.



Treemap of side-channel attacks identified in the SLR. Each color represents an attack class that targets different items. Figure 10.2

Side-Channel Attacks in Literature 10.3

Initially, two research questions were asked that will be answered in this section. We will take a closer look at what side-channel attacks do exist in the literature and what they are targeting. Then, the second question considers what the attacks have in common and how they differ. Successively, a cluster analysis is presented to help understand attacks according to the characteristics presented.

Overview of Attacks (RQ1) 10.3.1

In total, five different kinds of side-channel attacks can be identified from the literature that specifically use a smartphone. These are *fingerprinting*, *keylogging*, *location inference*, *user knowledge*, and *covert channel*. Three of the five classes contain different attack targets, which allow a more granular analysis. It is striking that keylogging attacks are the most common with 51 % of all attacks. This is followed by user knowledge (with 20 %), location inference (13 %), and fingerprinting (11 %). The creation of covert channels is less frequent with 5 % compared to the other ones. Figure 10.2 illustrates the distribution of the side-channel attacks categorized by target and attack class.

Table 10.2 Assignment of the individual publications to the respective attack class. The dominant class of attacks is keylogging with most works addressing this topic..

Attack Class	Publications
Covert Channel	257 [†] , 258 [†] , 20 ^{‡,*} , 36 [‡]
Fingerprinting	106 [†] , 429 [*] , 94 [†] , 423 [†] , 93 [*] , 95 [†] , 293 [†]
Keylogging	60 [†] , 371 [*] , 140 [*] , 26 [†] , 334 [†] , 355 [‡] , 345 [*] , 173 [†] , 275 ^{‡,*} , 265 [†] , 313 [*] , 251 [†] , 409 [†] , 208 ^{‡,*} , 299 [†] , 41 [†] , 232 [†] , 262 [†] , 79 [*] , 428 ^{‡,*} , 352 [†] , 339 [†] , 366 [‡] , 394 [*] , 288 [†] , 22 [*] , 272 ^{‡,*} , 338 [†] , 159 [*] , 192 [†] , 353 [†]
Location Inference	172 [‡] , 181 [†] , 163 [†] , 276 [†] , 231 [†] , 426 [†] , 47 [†] , 277 [†]
User Knowledge	11 [†] , 96 [†] , 141 [*] , 412 [†] , 21 [†] , 45 [†] , 152 [†] , 67 [‡] , 259 [†] , 280 [†] , 27 [†]

* Uses multimedia: camera, microphone

† Uses IMU: accelerometer, gyroscope, magnetometer

‡ Uses environmental: barometer, ambient light, temperature

We now briefly introduce each attack class. Each paper is assigned to a dedicated attack class. An overview can be taken from Table 10.2.

Fingerprinting

First, **fingerprinting** attacks aim to generate a unique identifier to, for example, *track* a single device over time or link different actions by a single user. This makes the need for system-side unique device identifiers obsolete, especially since strict permissions usually protect them. Sensor-based fingerprinting attacks are based on the specific characteristics of a device and its composition. This includes mobile devices that distinguish themselves not only from installed hardware but also from usage-based distinctions [293]. Even devices from the same series are subject to these slight variants, which do not affect the proper functionality. Zhang et al. [423] exploit the calibration of IMU sensors that are performed by the manufacturer or by software to generate a unique id with an entropy high enough to uniquely identify every device. Their attack works on Android and iOS. Using motion sensors from IMU seems to be feasible and was often seen in literature [94, 95]. Other approaches also exploit imperfections in the manufacturing process that affect e.g. the audio output [93, 429] or sensor accuracy [106]. Another approach uses the physical properties of electronic components, depending on the type, age, or even the generated magnetic fields, all of them that have enough entropy for fingerprinting via magnetometer [293].

Next, **keylogging** attacks represents with around 50 % the majority of all sensor-based attack publications and is, therefore, the most represented class of attacks. Such attacks use either the smartphone's keyboard or intercept keystrokes from external keyboards to obtain sensitive data from a user. On the one hand, this data can be general text input such as messages, which can be the basis for further attacks (i.a. spearphishing). On the other hand, attacks specifically target credentials such as PINs or passwords. Most attacks target PINs that include the device PIN or the graphical pattern [428] that protects the device (a common method in Android) [22, 26, 41, 60, 265, 288, 338, 345, 352, 353, 355, 366, 394]. Other approaches are focused on recognizing and extracting specific data, such as credit card numbers [313]. Additional work extends this kind of side-channel attack to even extract full text [299]. They do so by first roughly mapping accelerometer and gyroscope data to tap events and then refining the derived characters using language models to generate a meaningful text. In general, mainly ML-based classification and clustering approaches are used, which usually process the raw data transferred into feature vectors. However, even though most models do not need client-side training by the victim, labeled data is often needed for training purposes. Most attacks use an accelerometer or gyroscope to access the data entered via the keyboard. Wang et al. [394], however, use the front camera to track a victim's eye movement while entering a PIN and thus calculate the likelihood for specific passwords. Keylogging attacks are also implemented as a client-server application. In this case, data is collected on the client side e.g. through a website that has access to motion sensors and is continuously forwarded to a server that performs the actual analysis (e.g. [262]). Some attacks use a full QWERTY keyboard [275] while others are limited to specific environments like pin patterns [428]. Three works in the field of keylogging attacks do not target the smartphone itself but use its sensors to derive information from another device in close vicinity. Two of these three works try to find the keystroke performed on another keyboard [251, 272], while one approach uses the microphone to gather high-pitched noise emitted during cryptographic decryption to extract an RSA secret key. Such an attack is also called "*acoustic cryptanalysis*" [140].

Keylogging

The third class of attacks aims to derive location profiles of a victim and is called **location inference**. We identified two distinct goals of attackers. First, attacks try to *reconstruct a victim's trajectory*. Attacks record sensor data to derive events. This includes events that have been defined in Section 5.4 in the context of the work. Curves [163], or their angles [231] are derived and compared with map data to determine the route in exhaustive- or heuristic-based algorithms. More sophisticated attacks include more sensors to also derive the direction using the

Location
inference

magnetometer to enable tracking in larger areas [276, 277]. The accelerometer is used in most attacks [163, 231, 276, 277] with the exception of Ho et al. [172], whose attack is based on the barometer and compares pressure time series data with topographic elevation data and road maps for a given region. The attack surface, i.e. the area that can contain the trajectory, varies between attacks; for example, Ho et al. [172] uses areas of 90 km², while Li et al. [231] uses up to 900 km², although the success rate decreases significantly for such high areas. In Chapter 12, we present an improved attack for location inference that achieves an identification rate of almost 9 out of 10 routes for areas as large as 4500 km². The other 38 % of location inference attacks try to *pinpoint* a user to a given area, which can be the position within a building [426] or a metro line in one case [181]. The attack of Zheng and Hu [426] derives specific characteristics such as walking stairs or taking elevators to determine the position. This requires extensive knowledge about the target building. The same holds for the second attack, where a metro sensor pattern has to be learned previously. Some attacks to pinpoint users require additional hardware, although the application only uses permissionless sensors [47]. Altogether, location inference attacks underly specific constraints like the knowledge of the rough area of a victim's position or even prepared environments to be successful.

User
knowledge

User knowledge as a class of attacks has different objectives. One is to obtain information about a user's *media usage*. These attacks make up the majority of user knowledge attacks with 50 %. For example, information about websites visited via magnetic fields emitted by the CPU [259] or the apps used [259, 280] should be inferred. Both attacks rely on learned fingerprints and operate in closed-set environments. Furthermore, not only are objects on the smartphone of interest but smartphones are also used to derive knowledge from devices in the environment [45, 67, 141]. Attackers also want to eavesdrop on spoken words *reconstructing audio* from sensor data collected by an accelerometer or gyroscope. The goal can be to identify a specific *characteristics* of a user, such as his ethnicity [11] or his age [96]. Furthermore, simple words or even conversation reconstruction approaches [21, 27, 152, 412] were found in 33 % of user knowledge attacks. Interestingly, Griswold-Steiner et al. [152] measures facial movements to allow conclusions about the words spoken, as well as about the person speaking.

Covert
channel

Last and less present in our document corpus are **covert channel** attacks that target to enable data exchange between entities that are not intended to communicate due to security policies or similar. The *communication* is hidden and may not be detected by security mechanisms because it does not use typical communication methods of the systems. Bartolini et al. [36] present an attack that uses the

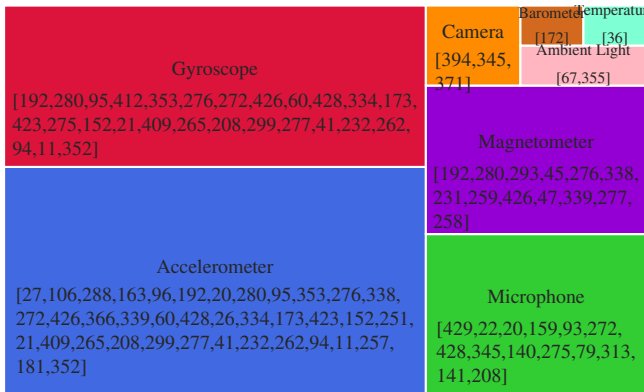


Figure 10.3

Distribution of sensor for a specific attack. The most used sensor is the accelerometer followed by the gyroscope. Rarely seen are sensors such as barometer, temperature and ambient light. Also, permission-protected sensors are found in literature presenting side-channel attacks.

temperature sensor to allow communication between two cores in a multi-core environment with one CPU having privileged access to communicate. It is called a sink and receives information (with a low data rate) from a communication-restricted core. Another approach uses the ability of the accelerometer to detect vibration [20], for example, of low-frequency sounds emitted by the speaker of a device to bypass information from private browsing sessions [257]. A similar approach also attempts to deduce information from private browsing sessions by exploiting disturbances in the magnetic field induced by specific CPU loads [258]. Only 5 % of the identified attacks fall into this category, two of them explicitly targeting private browsing sessions.

Structure of Attacks (RQ2)

10.3.2

We now answer the second research question and elaborate on the structure of sensor-based side-channel attacks. We focus on the attack vector, type of attack, attack frequency, and sensors involved.

After the previous question supported the identification of side-channel attacks in the literature, the next step is to ask how these attacks are carried out. Of interest here is which attack vector is used with the attack vector being defined as the specific path that is exploited to execute one of the five attack classes. The attack vector is a multistep process in some cases. The first superordinate path

Attack vector

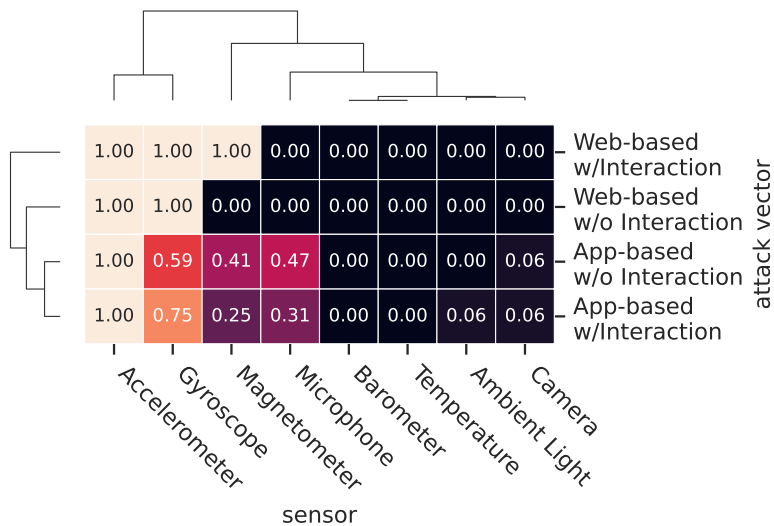
that has emerged is the distribution, as introduced in Table 10.1. On the one hand, attacks can be carried out via an installed application; on the other hand, websites also represent a path. Attacks carried out through websites only have to be called by the user. The necessary step of a setup in the form of an installation is omitted and thus represents a lower hurdle. In addition, applications can continue to run in the background to collect sensor data further, while websites rely on the browser. In addition, APIs to access sensor information via HTML5 were drafted in 2011 [383] with broad compatibility for these new interfaces needing to evolve, while the Android Software Development Kit allows accessing this information from Android 1.5. In the second step, some attacks require user interaction. Interaction can consist of the user having to interact with the attack (or the application). For example, ML models must be trained [355, 409] or certain procedures must be performed [47]. In other cases, certain preconditions are necessary, such as placing the device on a table so that ambient angles can be recorded accordingly [21].

Active and
passive

Interaction is not to be confused with active or passive attacker models. Most of the attacks are passive. According to the well-known definition of side-channel attacks, they collect information that does not correspond to the actual function intended by the manufacturer. Most attacks interpret correlations between sensor data and user behavior. For example, a passive nature keylogging attack is presented by Gupta et al. [159]. The authors use signal processing techniques to derive a set of likely tapped characters from sound recorded by the built-in microphones. They ultimately try to derive probable words and sentences from these tap sequences. In contrast, an active keylogging attack picks up that idea but emits acoustical signals while a user is typing. The sound is altered by a user's fingers, and the reverberation is gathered using the built-in microphone. The reverberation then allows deriving potential pin entries [79].

Relationship

Taking a look at the cluster map shown in Figure 10.4, a more detailed analysis can be performed regarding the execution of the attack in terms of the attack vector and the sensors exploited. A cluster map combines a heat map with a dendrogram that allows us to identify specific similarities. The cluster map is normalized on a row-level to understand the probability of employing sensors at a specific kind of attack vector. The dendrogram shows that application-based attacks indented on the level of interaction are close in terms of sensors applied. Interestingly, this also holds for web-based attacks that do not require interaction. Web-based attacks that require interaction tend to differ in their structure [258, 262]. It becomes evident from a sensor perspective that the accelerometer is the preferred sensor across all attack vectors. The high of a dendrogram bracket indicates how fast clusters denoted by a bracket are joined,



Cluster map illustrating the used sensors by a specific attack vector. Values are normalized for each vector. Figure 10.4

i.e. how close they are to each other. The barometer, temperature, ambient light sensor, and camera are promptly clustered since their bracket is also identical. The accelerometer and gyroscope are different from the set of other sensors, which may state that attacks are using only one or both of them. At the same time, an accelerometer, a gyroscope, and a magnetometer are commonly used, while other environmental sensors (barometer, temperature, and ambient light) are uncommon in either application-based or web-based attacks. It is also worth looking at which sensors are used for which attack in particular. The relationships are illustrated in Figure 10.5 in the form of a Sankey diagram. Shown are the sensors on the left, the five attack classes in the middle, and the top level of the attack vector on the right.

One can clearly see from Figure 10.5 that keylogging attacks make up the bulk. One can speculate that this side-channel has been known for some time and is therefore common. Such an attack based on IMU has already been presented 2012 by Cai and Chen [60]. Furthermore, the interest in sensitive data that is vulnerable to keylogging attacks can serve as motivation. Most keylogging attacks use an accelerometer, a gyroscope, and a microphone as sensors, though it should be noted that no attack that uses a microphone is web-based. Only

Attack frequency

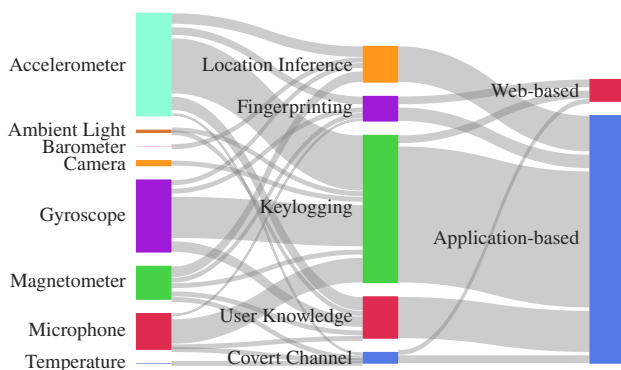


Figure 10.5 Sankey diagram illustrating the relation between sensors, attack classes, and attack vector. It is evident that keylogging attacks are the most common attack class. A majority of attacks is performed via a dedicated application.

two attacks that use a combination of an accelerometer and a gyroscope are web-based [262, 352]. User knowledge attacks and location inference attacks are found in 18 and 14 publications, respectively. Both attacks are based solely on the application. This is probably because user-knowledge and location-inference attacks need to collect data over a period of time. For example, location inference attacks use the magnetometer to detect corners while driving and then try to reconstruct a trajectory [231]. This data has to be collected continuously, and hence, web-based approaches might not be feasible. Fingerprinting attacks are found in 10 publications and use an accelerometer, gyroscope, microphone, or magnetometer, although an accelerometer and a gyroscope are the most common. Attacks of this class are evenly distributed between application-based and web-based approaches, with some approaches also being able to work in both environments [423]. This may be related to the fact that short periods of data are sufficient to create a unique fingerprint [423]. Covert channel attacks are relatively rare, with only three observations. All of them are instead starting in 2016 [36]. Two of them are application-based.

Sensor Appealing are sensors of the class of motion sensors (i.a. collected using an IMU) because they do not require the victim to allow their usage (e.g. [394]; c.f. Section 3.1). This is also true for environmental sensors (ambient light, barometer, temperature). As a consequence, accelerometers, gyroscopes, and magnetometers are often used. Striking is the microphone, which is permission-protected but commonly used for keylogging attacks since it can detect keystrokes on internal

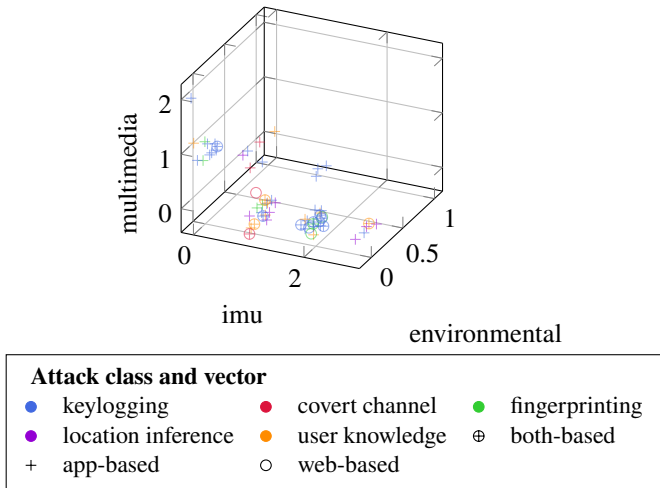


Figure 10.6

Clusters formed by the number of involved sensors. Each shape represents a publication with colors illustrating the respective attack class. It is evident that most attacks are based on the IMU. Attacks using environmental sensors are rare.

or external keyboards. Environmental sensors are rarely used in only 7% of all attacks. This may be due to the fact that these sensors allow little reference to a subject. The camera is used exclusively for keylogging [345, 371, 394]. However, we find that sensor fusion is used in 30 of the 61 attacks presented.

Clusters

10.3.3

Figure 10.6 illustrate clusters of attacks. Each color represents an attack class and symbols the attack vector, i.e. if an attack is executed at an application level or on the web level. Understanding clusters and involved sensors may also help to derive protective measures. The next section focuses on methods to circumvent specific attacks. Specific clusters are visible when approaching the figure from an attack-class perspective.

First, we can see that the keylogging attacks are distributed between sensors. Most attacks are based on IMU sensors independent of their type or target. In particular, only keylogging attacks and one covert channel attack combine IMU sensors with multimedia sensors, but no attack uses a combination of environmental and other sensor classes. Location inference attacks tend to use more sensors from the IMU which is especially true for more recent and

Attack classes

sophisticated attacks with up to three [277, 426]. 67 % of all attacks specifically use motion sensors (IMU).

Attack vector The dimension of the attack vector is different. It can be seen that most web-based attacks are based on IMU sensors, almost not using sensors of the environmental or multimedia dimension. This might be related to how the accessibility level of HTML5 APIs. There is no standardized API that allows access to environmental sensors (barometer, temperature). This excludes the ambient light sensors that are only used in two user knowledge attacks that are also application-based on the other site.

10.4 Outlook on Protective Measures

The structure of side-channel attacks allows us to conclude some countermeasures that may prevent an attack entirely or at least decrease its performance. However, multiple methods have to be combined to provide an adequate protection level depending on the attack's composition.

Inappropriate permissions We already discussed specific permission-related constructs in Section 3.2. First of all, some attacks can be prevented in current systems with *permissions*. This is true for attacks that employ environmental and multimedia sensors. Both sensor classes are already protected with specific permissions that ultimately protect against side-channel attacks such as Simon and Anderson [345] that perform eye tracking while a user is typing. For example, there is no need to permit a keyboard to access the camera. It is also achievable for users to employ their methods to protect against side-channel attacks, specifically for multimedia sensors. For instance, a subject can cover a device's camera or microphone, ultimately reducing a privacy violation risk. This can act as a second safeguard even in cases where an application has authorized access to the camera or microphones but should not use them at some point². Similar approaches may also be used in combination with the ambient light sensor.

Respecting process state Also, separating sensor access in the foreground and background tasks may increase the level of user protection since most applications, especially location inference attacks, continuously collect sensor data in the background [231]. This may not work for attacks that require small amounts of data that can be collected in a short period of time [423], e.g. within the time-out interval of Android switching an app from foreground to background.

²A communication app can use the microphone and camera when a user is on a call but shall not use it otherwise.

However, side-channel attacks may be hidden in decoy applications [231], making it difficult for a user to comprehend the possible misuse of data. Furthermore, it may not be applicable to separate a function from an attack. This is the case for privacy-relevant models, such as the model presented UBI. Methods for *altering data* may be applied to balance the need for privacy and the requirements of integrity and plausibility. In principle, classical methods can be applied at this point, as they are known, for example, from database security. On the one hand, this includes the generalization of data (e.g. using rounding [423] or lower sampling rates [288]) or the insertion of noise [95]. However, this is critical w.r.t. the application used. As shown in Chapter 2, use cases are defined by the quality of sensor data and assume a certain quality, which is artificially lowered for protection. Furthermore, we show in the course of the work (see Chapter 14) that sophisticated ML models are robust to changing data, especially in multivariate time series scenarios. Nonetheless, falsifying data may work in fingerprinting attacks since unique patterns (such as the gain matrix in Zhang et al. [423]) may be destroyed.

Falsification
of data

Protection against covert-channel attacks is difficult to achieve since some attacks are executed on a second device controlled by an adversary. Therefore, using permissions or similar concepts on the target device is not feasible. In the case of Anand and Saxena [20], the device loudspeakers play a low-frequency sound. Playing audio is considered the loudspeakers' main task and may occur in multiple scenarios (watching a video, listening to music, playing a game). Recall that side-channel attacks use methods in a way that is not intended by system design. Hence low-frequency sounds may be considered an anomaly since the emitted frequency may be inaudible. It conflicts with the intention of a loudspeaker to transport an audio signal to a user and may not decrease any convenience if such sounds are filtered by a security engine such as OS itself. However, detecting anomalies is difficult to achieve and challenging to balance, as users might be sensitive to misdetections and functionality limitations.

Interception
of
conspicuous
algorithms

Especially keylogging attacks employ multiple sensors to generate accurate predictions to extract keystroke data from sensor readings. Sensor fusion is then performed on an array of data. Previous approaches may not work due to the sensors and proposals involved. Therefore, the literature proposes to disable access to the sensors at all while users type [41]. This can be considered a rather strict approach that may impact the user experience. Some keylogging attacks work using heuristics or pre-learned ML models that predict a keystroke from sensor readings. When the layout of the keyboard is dynamically changed with each activation [355], the ground truth and assumptions of these models and heuristics no longer represent the situation. Hence, accuracy will likely decrease.

Context-
based sensor
deactivation

However, users may not accept such a radical protective method as the habit of typing will suffer.

Summary In summary, there are challenges in addressing side-channel attacks. As the abundance of the literature shows, existing methods for protecting sensor data are not sufficient. Thus, a clear threat to security, but especially to users' privacy, can be attributed to sensors. Only a combination of protective measures and improved user awareness can support a safer environment. In Chapter 13, we give an outlook on current frameworks to protect against side-channel attacks but also address privacy issues in data sharing. They help users manage their data and individual privacy needs. In Chapter 14, we present a technology that can increase the privacy level of sensor-based applications for the user while ensuring the integrity of the data that is required for the meaningful evaluation in business models.

Having seen that multiple side-channel attacks are possible in the area of sensor data, this chapter explores and presents a meaningful proposal to identify drivers solely from unprotected sensor data (c.f. Chapter 3). Despite using leaked information to tell a driver, they are meaningful use cases. Furthermore, using the information from an IMU is cost-effective and easy to implement since no additional hardware is needed within a vehicle. Examples of added value by identifying a driver include, but are not limited to:

- ▶ By identifying a driver, it is possible to exclude a specific person from a set of legitimate drivers and, therefore, detect car theft. This information can then be used to inform the car owner or insurances [58, 189, 250, 254].
- ▶ Detecting the driver of a car allows to set user-specific settings in a comfortable way, such as favorite radio station, personalized temperature settings, or recommend location-based services (e.g. restaurants) explicitly tailed to the driver's preferences [121, 162].
- ▶ In the context of electric mobility, identification of the driver can help provide better estimates of the remaining range or predict energy consumption with greater precision [247].
- ▶ Bus or taxi companies can use the information of a driver to detect misuse; for example, if a person without the required training is using a specific vehicle, he ordinarily is not allowed to use [58].

The listed examples are primarily beneficial and might be appreciated by an individual. However, there are specific use cases where one party has a disadvantage or at least a limited interest in being identifiable. The running example

Acknowledgement

Parts of the research presented in this chapter are based on work published previously [322] and supervised work [S4].

of PHYD might be relevant. Car insurance companies or other stakeholders can differentiate between multiple drivers and thus enforce their contracts or policies for i.a. a specific number of drivers. They may have the ability to request a penalty charge, terminate a contract, or be only notified about the incident [58, 121, 189, 365]. In the event of an accident, it is possible to identify the driver at that particular event and therefore collect evidence [121]. Furthermore, it may be realistic to analyze whether the driver was under the influence of alcohol or drugs if his driving styles differ from his standard behavior. The given use cases separate themselves from the more trivial approach of driver classification (see Sidebar E).

Since driver identification can be beneficial or disadvantageous, our presented approach can be considered an attack if data is collected in a nonobvious, non-communicated way but can also be used for the given examples by intention.

Sidebar E Driver Classification vs Driver Identification

Classifying a driver is the task of assigning a driver to a specific, arbitrary category. The most common approach is to classify an individual's driving behavior and distinguish between aggressive, neutral, and passive drivers. The output of a classification process is always a (most of the time exclusive) category. This information may be relevant for e.g. insurances. Synonyms for classification are categorization, association, or recognition. Classification is not focused in this SLR, however, Marina Martinez et al. [248] surveyed on this specific topic.

On the other hand, identification attempts to recognize and name a specific driver from a set of drivers. This task is much more demanding than classification because, especially for larger groups, the derivations in the input data between multiple drivers may be pretty small. Therefore, papers either use a closed-set or an open-set approach. Closed-set can be considered to be easier since the number of drivers is and their driving style is known in advance (e.g. [58]). However, open-set approaches use unknown drivers not included in a potential training set by keeping all requirements of closed-set approaches. Open-set approaches may be needed for e.g. theft detection [254]. Most of the publications found use closed-sets.

It should be noted that the terms classification and identification are often used in a fuzzy way, and thus it is not possible to use the term only. In particular, it is important to analyze the context of the terms. We concluded that the isolated

term “*driver*” in combination with “*classification*” is in the field of identification [421] while “*driving style*” together with “*classification*” is associated with a classification problem [358].

This chapter presents the first leg of our attack tuple with an identification approach particularly tailored to the scenario of PHYD using only zero-permission sensor data comparable to a data processor. To the best of our knowledge, this is the first approach using k-Nearest-Neighbour (kNN) with Dynamic Time Warping (DTW). We contribute with

Contribution

- ▶ an in-depth survey on current approaches to driver identification with a focus on feasibility in the field of UBI,
- ▶ a newly designed side-channel attack for driver identification that is different from previous approaches as it relies on a distance-based algorithm,
- ▶ a full evaluation of the approach to understanding the performance and potential limitations, and
- ▶ an outlook on how the privacy question of UBI is more severe in the real-world based on assessments of an extended identification attack.

It should be noted here that in Chapter 14 of this work, we also present countermeasures for protection against attacks of the user knowledge class, specifically in the challenging context of UBI.

In this chapter, we first perform an extensive SLR to get an overview of the approaches for driver identification in Sections 11.1 and 11.2 as it is performed in the literature. Then we reflect on whether the current approaches are feasible in the given context of PHYD. Eventually, we present our approach based on ML to perform driver identification in Section 11.3. A thorough evaluation using data sets presented in Chapter 5 is performed in Section 11.4. Section 11.5 then projects the identification approach on a more realistic scenario and discusses its feasibility, before concluding the chapter in Section 11.6.

Structure

Structured Literature Review

11.1

In the following, the research questions are introduced as well as an overview of the document corpus that is identified within the search process.

11.1.1 Research Questions

The primary question that should be addressed in this study is as follow: “*What is the state-of-the-art in driver identification?*”. Because of the complexity of this question and its many aspects, one needs to answer, we formulate three subordinate questions: we first try to find motivations for driver identification, afterward and because of its empirical nature, we address this topic in the first two questions, then analyze their methodology, and later look into the results in a fourth question:

- Q1** Considering the data-driven topic, what are the inputs for driver identification, and which acquisition methods researchers employ?
- Q2** Is there a typical or standard workflow for identifying the driver? Are there substantial differences in the approaches?
- Q3** What methods do researchers use for the actual identification process? How do they perform?

We selected these research questions because we want to analyze the entire driver identification process performed in the literature. This section includes work by Gihl [S4] who performed a SLR on the identification of drivers.

11.1.2 Search Process

We started our literature review by analyzing the following databases: IEEE Xplore, SpringerLink, and ACM Digital Library (*forward-search*). Furthermore, we searched outside of those databases, in particular, for references to relevant articles (*backward-search*). Additionally, Google Scholar was also used to identify work that is potentially appropriate for our research questions. However, a significant overlap of these results with the already scanned resources occurred. To capture relevant work, we used different combinations of keywords, namely¹:

```
( DRIVER & ( IDENTIFICATION | RECOGNITION | AUTHENTICATION
  ) )
& ( SENSOR | SMARTPHONE )
```

Explicitly, we skipped the related term “*classification*” because it is outside the context of this survey. However, the placement of this word is often confused with “*identification*” (c.f. Sidebar E).

¹For an explanation of the notation, see Appendix B

The search process was a manual search of the said terms. We did not limit ourselves to specific conference proceedings or journal articles, but we included everything matching the search terms in the first collection. However, to grasp only recent developments in driver identification, we limited the search to include only work since 2013 with one exception. The exception from 2007 was included since it was cited extensively in the recent work we identified. Therefore, we included it because of its apparent importance.

The feasibility of each potential article was then analyzed by reviewing the title and abstract. Unsuitable work not directly related to the field of identification was then removed, and all other articles were prepared for an in-depth analysis where the full text was scanned. Therefore, we scanned the remaining data set for our inclusion criteria.

1. Firstly, the paper targets topic identification, not classification or association. These terms were often used in a fuzzy way.
2. Secondly, the identification of the *driver while driving* a vehicle is focused, though some workers tried to tell the difference between the driver and co-driver.
3. Also, we eliminated work whose research methods were unclear or not detailed enough fully grasp the identification process.
4. Lastly, we culled all work that does not use sensor data closely related to vehicular environments. In particular, we did not include papers that used vital sensor data e.g. from a potential driver.

The inclusion criteria were then applied to the articles found and discussed between two researchers to arrive at a decision. This was especially the case where the terminology of identification and classification was somehow unclear because the inclusion criterion was not always trivial to implement. In addition, only peer-reviewed publications in English were considered.

Relevant findings

11.1.3

From the initially 50 articles found, we were able to identify 20 relevant publications (including one poster) which discuss or are closely related to the topic of driver identification using device agnostic sensor data. We were able to reduce the number of papers from 50 to 22 after scanning the titles and abstracts of the articles. Afterward, two papers were removed after having read the complete text, resulting in the final 20 papers Table 11.1 is presenting. Interestingly, we

were unable to derive any increasing or decreasing interest in the community with an unequal publishing rate throughout the years during 2013–2018, the peak being in 2016 with eight publications, followed by 2018 with six papers. One work [267] has been included even though it was not published in a peer-review format as it poses a meaningful proposal that, according to the author, fits the scope of this review.

Table 11.1 Overview of the 20 publications identified in the SLR. The works are assigned to different disciplines.

Publication	Year	Publisher	Field
Tahmasbi et al. [365]	2018	ACM	Network
Marchegiani and Posner [247]	2018	IEEE	Vehicular
Jeong et al. [193]	2018	IEEE	Vehicular
Jafarnejad et al. [190]	2018	IEEE	Vehicular
Gahr et al. [135]	2018	IEEE	Vehicular
Bernardi et al. [43]	2018	IEEE	Machine Learning
Virojboonkiate et al. [381]	2017	IEEE	Network
Jafarnejad et al. [189]	2017	IEEE	Vehicular
Fung et al. [133]	2017	IEEE	Health
Zhang et al. [421]	2016	ACM	HMI
Yang et al. [413]	2016	IEEE	Network
Martinez et al. [254]	2016	IEEE	Vehicular
Markwood and Liu [250]	2016	ACM	Computer Science
Kwak et al. [223]	2016	IEEE	Computer Science
Hallac et al. [162]	2016	IEEE	Vehicular
Enev et al. [121]	2016	De Gruyter	Computer Science
Burton et al. [58]	2016	IEEE	Network
van Ly et al. [376]	2013	IEEE	Vehicular
Quek and Ng [304]	2013	–	Machine Learning
Miyajima et al. [267]	2007	IEEE	Computer Science

Looking at the result, it became evident that the most relevant publisher for this kind of topic seems to be IEEE, with the IEEE Conference on Intelligent Transportation Systems being the most present platform for such work. Furthermore,

most of the remaining work was published in the context of conferences/journals focusing on vehicular environments, only some of them from other fields such as Machine Learning, computer security (including privacy). Health, Human-Machine Interaction and networking-related formats were seen sporadically.

Analysis of Driver Identification Methods

11.2

In the next section, an in-depth analysis of driver identification methods is performed that is based on the identified document corpus.

Data Acquisition (RQ1)

11.2.1

The identification of a driver using his driving style is a data-driven approach. Consequently, we now analyze the context in which the data are gathered and what specific attributes or sensors are used.

Experimental setup

Sensor data is recorded in real-world scenarios or isolated simulator results are used. Figure 11.1a illustrates the different scenarios and data sources. Almost nine out of ten publications use data gathered in real-world scenarios. However, there is no relation to the publication year; also, newer publications [58, 413] rely on the usage of simulation results without disclosing the respective software in any case. Some articles also use data of both types, for example Miyajima et al. [267], although their approach was not generalizable since the simulator did not have data on pedal pressure, while real-world data did not provide information about the distance to the previous car. This shows that almost all approaches are tailored towards the scenario. Some articles used publicly available data sets such as UYANIK [189, 190, 254] and others [133, 267, 304] but most of the authors decided to collect their own undisclosed data. This is, in particular, true for the source of the data. Data derived directly from a vehicle, e.g. through OBD [135, 223], is found in 71 % of all real-world articles. External devices (10 %) are, for instance, Raspberry Pis with accelerometers [381]. Three publications [250, 365, 421] also use smartphone data (recorded through self-developed applications); however, they are subject to explained error sources (c.f. Chapter 2). In particular, orientation in the vehicle is important, but most authors were not specific about this fact, apart from “*placed in the central console*” [421] or “*almost all data was collected with the device sitting in the passenger seat, which in modern cars has a very small incline when looking forward, so the y-acceleration trace would commonly be reduced by a scalar constant so that the average would be*

Different
collection en-
vironments

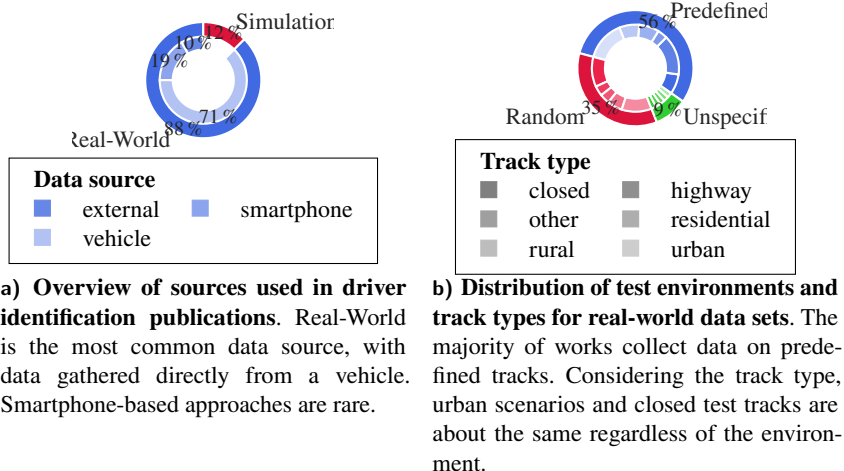


Figure 11.1 Quantitative metrics of the data sets that are used within the document corpus.

close to zero” [250]. All works assumed a fixed position (across test subjects). Some articles combined vehicle data with external data [133] or smartphone data [421].

Trajectory
shapes

Analyzing the data sources in more detail yields the result that half of the publications use predefined routes to collect the data next to random trajectories. Figure 11.1b depicts the distribution. Notably, simulations are always using predefined test tracks where the authors prescribe a route to be followed by the proponents. This type was also found in real-world data sets. Predefined test tracks included drives on public roads [190, 223, 247] or maneuvers in a parking lot [121, 223]. Hallac et al. [162] is based on random experiments in which participants drove a day independently with a test vehicle while daily drives are used in Zhang et al. [421]: They recorded routine tracks from three couples in their cars for four days. Additionally, each category, predefined or random, can be broken down into different types of roads. Urban roads are the most common, as they usually produce more events to be analyzed than, for instance, highways.

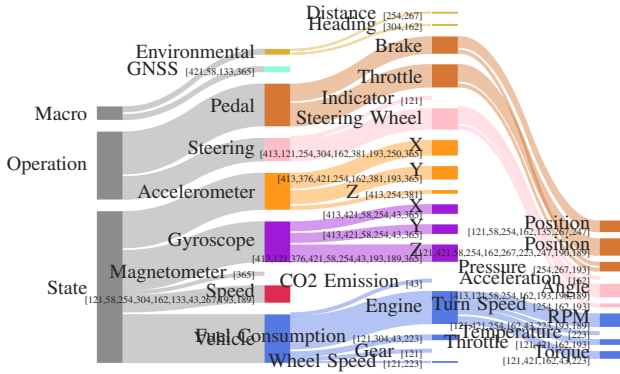


Figure 11.2

Overview of sensor data used in different publications.

W.r.t. the entropy of cars, both approaches are used in literature. People used multiple cars [43, 421] or a single test vehicle [121, 190, 247], although this does not indicate that the data set presents only a single vehicle (e.g. people used their car throughout the test [133]).

Number of vehicles

Sensor data

Depending on the data source used, different sensor readings are available, which directly impacts the identification process. Figure 11.2 shows an overview of all sensors extracted from the list of publications. It should be noted that only sensors eventually used for the identification process are listed in the figure, but some papers could be more clear regarding the data used (some papers listed multiple sensors whose usage in the identification process was inconclusive). Furthermore, some works derived additional information from sensors [190, 247, 421]. We also generalized some sensors due to very few occurrences [43, 121, 223]. Fung et al. [133] and Quek and Ng [304] derived the speed from GPS data, which can lead to some unstated inaccuracies. Markwood and Liu [250] in addition to GPS used an accelerometer.

Sources of sensor data

Sensors are organized into three classes, as Figure 11.2 shows. First, *macroscopic* sensors are less common but include work that uses, for example, a vehicle distance [254] to other traffic participants. We call that category macroscopic since it requires a second party to be conclusive. In addition to that class, *operation*-type sensors include data that is related to the interaction with input devices of a vehicle, i.e. steering wheel, turn indicators, or pedals. Steering

Three sensor classes

wheel operations yield promising results [121, 162, 193]. The last sensor class represents data on the vehicle state. This includes sensor data collected by the IMU. The magnetometer, however, is only found in one publication [365], probably due to reasons explained in Section 2.3 such as susceptibility to disturbances. The accelerator and the gyroscope are found almost equally, but the axes used are different. For example, a gyroscope was often used to detect and measure turns using gyr_z while an accelerometer was used to measure lateral (acc_x) or longitudinal (acc_y) forces. This underlines the feasibility of the accelerometer and gyroscope for driver identification [121, 133, 162]. The vehicle state also includes data from the vehicle itself, namely, the characteristics of the engine (e.g. RPM or torque), the fuel consumption, or the emissions. This data is only found in works that use the vehicle itself as a data source and extract them via OBD. As discussed previously, this data tends to have the highest frequency and quality.

11.2.2 Workflow (RQ2)

We know answer RQ2) that asks if there is a shared workflow between works and if there are substantial differences.

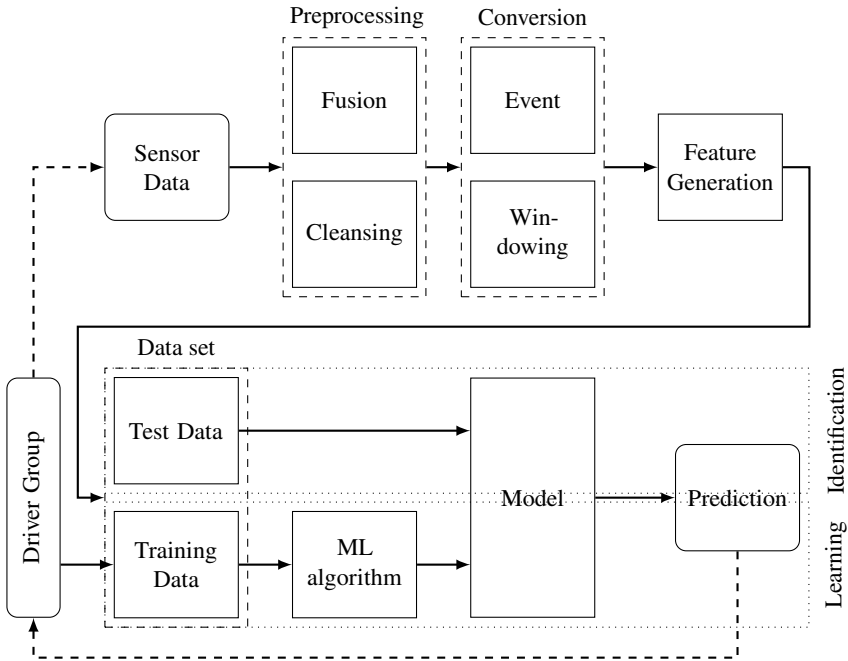
Meta-model

The identification of drivers follows a meta-model that was constructed after analyzing the document corpus. It is shown in Figure 11.3. We conclude that driver identification is a supervised task in closed-set environments. Closed-set environments are constructed by identification tasks in which each event is linked to a unique driver whose driving style is known during the learning phase. This is different from open-set recognition, where one needs to differentiate between an actual driver and a much larger class of unknown entities that have not been included in the learning process. Sensor data is gathered using methods explained previously, including applying multiple-attribute sensor fusion over time. Across-domain fusion is not found in any of the analyzed publications.

Preprocess-
ing

Several data cleansing steps are applied throughout the literature in a preprocessing step to handle different levels of data quality. They match the approaches already identified in Section 4.2.2.

- ▶ Either too short or too long records are discarded [133, 189, 190].
- ▶ Damaged records are disposed of, or too noisy readings were removed from the records [189, 190, 250]. Noise is reduced using different techniques, such as wavelet denoising [121, 162], median filtering [413] or low-pass filtering [190, 365].



Meta model of driver identification process. The literature considers the task of driver identification a supervised learning problem. Approaches differ in how the feature vector is constructed (either based on events or windowing). (based on Markwood and Liu [250])

Figure 11.3

- ▶ A sanity check is performed to remove records with too much missing data or incorrect data [43, 304, 381].
- ▶ Redundant sensor data is deleted (there is no information about conflicting data) [223, 254].
- ▶ Data set size is reduced by aggregating multiple values [162].

The application of multiple cleansing methods implies that there has to be domain knowledge about the data source. The literature is sometimes unclear as to what indicates the need for a preparation method and does not mention any recommendation or support for decisions.

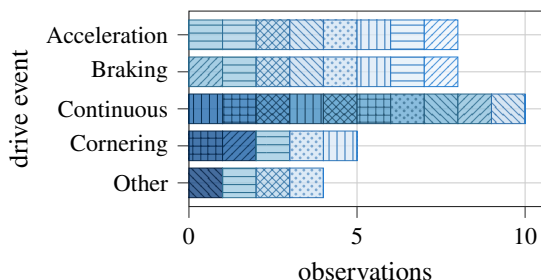


Figure 11.4 Overview of events used for driver identification. Each shape corresponds to a work in the document corpus. The identified elements include the events defined in Section 5.4. In addition, ten publications process a constant data stream and do not extract events. Four papers identify other event types.

Extraction Next, the continuous multivariate time series sensor data is converted to windows, as explained in Section 8.3.2. In addition to overlapping windows [121, 135, 189, 190, 223], there are also hanning windows [421] as well as static windows [43, 193] found in the literature. We found that only works that collect the continuous data stream perform windowing strategies. Event extraction also takes place. Events are similar to the definition in this work. This means they describe specific patterns in the time series data that can be matched to a specific driving activity. Figure 11.4 shows the distribution of events that were identified within the SLR. Each shape represents a specific work. Some use multiple events at the same time. In particular, acceleration, braking, and cornering are mostly found in works that do not rely on continuous data streams. Those works rather extract features from windows. The features will be explained consecutively. Furthermore, four works use unique events that are summarized as others. These include hand movement [413], velocity [304], pedal operation [267], and coasting [250]. The majority relies on acceleration and braking as events, sometimes in combination [133, 247, 250, 267, 304, 376, 381]. Jeong et al. [193] combined events and continuous data stream for driver identification.

11.2.3 Identification methods (RQ3)

The identification of drivers based on preprocessed data is subject to this section. Identification methods can be distinguished between features and approaches. Features are measurable properties that an appropriate approach can process. Consequently, the approach can perform the actual task of matching a set of

features to a driver. The matrix in Table 11.2 matches different classes of features with multiple classes of approaches.

Matrix illustrating the different approaches found in literature including features used for identification. It is evident that most works that apply machine learning use processed features while others use raw values. Table 11.2

		Algorithm					
		Event	Other	Raw	Spectral	Statistical	Unspecified
Feature	EL ¹	–	[365]	–	[189]	[189]	–
	ML ²	[135, 376]	–	[193, 223, 247]	[121, 421, 135, 162, 189, 247, 190, 254, 304]	[121, 421, 135, 162, 189, 376, 58, 223, 254]	–
	MM ³	–	–	[413, 247, 267]	[190, 247, 267]	–	–
	NN ⁴	–	–	[43, 193, 223]	[254]	[223, 254]	[381]
	O ⁵	[133]	[133]	[250]	[162]	[162, 133]	–

¹ Ensemble Learning ² Machine learning ³ Mixture Models ⁴ Neural Network
⁵ Other

Apart from feeding raw data, i.e. the preprocessed or incoming data stream, into an approach, most works reshape sensor data. Those features can be categorized into statistical, spectral, or event-based features. Most works use statistical features, including median, average, maximum, minimum, quartiles, standard deviation, autocorrelation, kurtosis, skewness, and duration [135]. Spectral features are frequency domain metrics and are often found in voice or sound processing. For instance, Zhang et al. [421] uses the power spectrum and cepstrum, including their entropy and standard deviation. Furthermore, event-based features are re-

Features

lated to events and describe meta-data of them, e.g. the duration of an event [133]. In addition, there are extraordinary (other) features such as the curvature of the path [133]. Virojboonkiate et al. [381] missed mentioning any features. We also found that some papers constructed a multitude of features for windows of up to 6,400 features for each window [421]. Combinations of features are common w.r.t. Table 11.2.

Classification
methods

Driver identification is a supervised task. However, different approaches to Artificial Intelligence are found in the literature. Most of the work apply an approach based on classical ML methods. These include Support Vector Machine [58, 121, 135, 162, 189, 193, 247, 304, 376, 421], Random Forest Classifier [58, 121, 135, 162, 189, 223] and kNN [58, 121, 135, 223] and Naïve Bayes [121, 135]. These are mainly combined with spectral or statistical features. It is noticeable that works use several methods simultaneously and compare their goodness against each other. For the identification task, Random Forest or Support Vector Machine seem to be the most suitable, depending on the work. Furthermore, ensemble learning approaches can be found that combine different learning algorithms to improve the results (c.f. [319]). These include AdaBoost [189] or Gradient Boosting [189, 365] (c.f. Section 8.5). Among the mixture models, we find work that employs the Gaussian Mixture Model [190, 267, 413]. This probabilistic model assumes that all data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters. Not all machine learning algorithms are able to classify other than two binary classes. Therefore, different strategies are used, such as one-vs-one [121, 190] or one-vs-all/one-vs-rest. This allows the usage of a wider field of classification methods. State-of-the-art cross-validation is used most of the time and yields better and more reliable results [223, 247, 376]. Another group of approaches employed for identification are NNs. In contrast to other approaches, raw data is the most frequently chosen class of features for this approach. For example, Multilayer Perceptrons [43], Convolutional Neural Networks (NNs) [193] or Extreme Learning Machines [254] can be found. Other approaches are seen only in specific works. For example, Markwood and Liu [250] uses a Kolmogorov-Smirnov statistical test. In particular, implicit assumptions such as the Optimal Velocity model are also included in identification approaches. This model is one of many traffic flow models [32] where the position to subsequent cars and therefore the distance between these cars imply the driven speed.

Results

Even though most works use identification accuracy as a metric to assess the quality of their identification algorithm, comparability between works is only possible to a limited extent. One main reason for this is the heterogeneous data sets as shown in Section 11.2.1. The identification approaches are hardcoded to

a scenario due to the features applied. Markwood and Liu [250], for example, places the identification in the context of vehicle theft. An example illustrates the non-comparability of the approaches. Jafarnejad et al. [189] and Jeong et al. [193] use a Support Vector Machine to perform the identification task. However, the first work achieves an accuracy of 90 % for a group of five drivers, while the other work only achieves 35 % for four drivers. Arguably, this may be related to features, data sources, or the type of tracks. Data sources are often not public. Therefore, no comparisons are made between approaches. Additionally, only a guess can be made which sensor data has the biggest influence on a driver's driving behavior and thus allows its identification. It can be stated that acceleration is listed as influential in some works [121, 133, 162]. In the end, no recommendation can be made. Sensors or features are often not examined individually. We also analyze the number of drivers used in the identification process. Interestingly, the number of drivers (called a classification group) tried to be identified differs from the absolute number of the driver in the data set: Most papers use fewer drivers in the identification process than the absolute number of drivers. For example, Jafarnejad et al. [190] considered classification groups of five, 15, and 35, while the data set contained 67 drivers. They randomly select the respective number of drivers and redo the evaluation ten times. Hallac et al. [162] selected twelve corners (i.e. the road element as identification was performed using this specific shape) for driver sizes from two to five from the set of 64 persons. However, they filter their data set to only select drivers that drove the same corners and only use the drivers with the most corner events. Zhang et al. [421] only listed the quality of their prediction per vehicle, i.e. drivers shared cars, ultimately resulting in smaller test sets and therefore potentially increasing the identification result.

Time Series-based Approach for Driver Identification

11.3

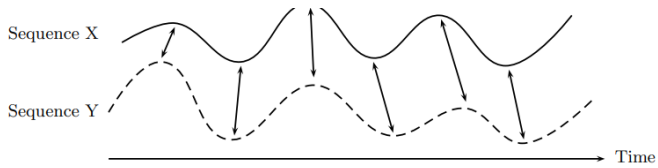
Section 9.4 introduced the privacy problem in the context of UBI. The reasons for the threat are various and are already discussed in Sections 1.2 and 9.4. We may assume that the current premiums and their respective processes are designed in line with the GDPR; however, as this chapter will illustrate, the regulations are still not sufficiently implemented. Let us take *data minimization* and *purpose limitation* as an example. The former may be present when only sensor data or limited events are submitted to a data processor. Then, side-channel attacks, such as the presented identification attack based on purposely published information, will undermine the requirement of purpose limitation. Both properties should be present in a system *by design* as we introduced in Chapter 6. We have shown that privacy and functionality can be realized in the same way in Chapters 7 and 8.

Adversary An adversary in this scenario is interested in identifying a driver *a posteriori*. A driver collects data in the PHYD context using a smartphone application or any other device equipped with a IMU. This allows an adversary to derive any needed information. Therefore, he can access \mathcal{M} as defined in this work (e.g. after uploading data as required by a potential PHYD premium). His ultimate goal is to assign a trajectory to a known person p from the set of \mathcal{P} . For instance, he wants to guess the driver of a journey out of the pool of people sharing a car. On average, a vehicle is shared among five people; therefore, we selected five people from this work’s data set and kept them constant throughout the experiments.

Approach overview In the following, the presented approach for identifying drivers will be placed in the scenario. The findings of the literature study serve as a framework. A conscious decision is made to use smartphones as a data source, as they are dominant in the UBI context, as shown in Section 9.3. This choice sorts out many sensor data that, in principle, is feasible, processable, and accurate in the context of driving identification, such as pedal operations [121, 193] or vehicle-specific data [43]. We apply event-based detection by using two complex events, namely acceleration and braking. These are also applied mainly in the literature, with features illustrated below. We chose to use raw sensor data that is processed in a time series manner using the DTW algorithm (see Sidebar F for a brief introduction). The data is then feed to a k-Nearest-Neighbour algorithm to predict the a concrete driver from the pool of drivers. To the best of our knowledge, this is the first approach to identifying drivers using a combination of Dynamic Time Warping (DTW) and k-Nearest-Neighbour (kNN). We will evaluate the applicability of the task using real-world experiments and data. An extensive evaluation will also analyze the impact of events on accuracy.

Sidebar F A brief explanation of Dynamic Time Warping

Dynamic Time Warping (DTW) is an algorithm to compare two time series of arbitrary length to assess their similarity ultimately. We refer to Müller [271] for this overview. It is used for its application to speech recognition to identify patterns even when one person may speak at a different pace than another person. However, it is also well-suited for data mining as it can handle deformations, for instance, probably due to different accelerations in sensor data recordings \mathcal{M} as they are found within this work. Still, it is possible to identify an underlying pattern that may be present in both sequences (e.g. a reference sequence such as a learned pattern and an actual recording).



Example of the alignment of two sequences X and Y. The alignment points are represented as arrows. [271]

Figure 11.5

The match between two length sequences (N, M) is performed under three restrictions to find an alignment that minimizes a cost measure. This alignment of two time series is considered a *warping path* (c.f. Figure 11.5). First, the starting and ending elements of the warping path are the start- and endpoints of the two sequences, respectively, so that both sequences are aligned. Second, the monotonicity of the sequences must hold as time series are chronologically ordered per definition in this work. Third, a step size condition must hold; that is, the distance between elements in both sequences that are warped along is limited so that no repetition (i.e. an element from a sequence is mapped multiple times) or skipping of elements occurs.

DTW's runtime $\mathcal{O}(N, M)$ which is due to the fact that the optimal warping path between both sequences has to be found. Multiple variants of DTW have been proposed that tackle the challenge of optimizing the runtime or the warping path.

Features

11.3.1

Events from Section 5.4 should serve as input data for our algorithm because they are easily collectible via a smartphone application. Furthermore, they are widely used in the PHYD scenario (c.f. Table 9.1) and thus are likely to have enough entropy to be unique or quasi-unique for a driver. However, this claim is subject to further investigation in this chapter.

In general, we use \mathcal{M} or more precisely \mathcal{E}_c found within a trajectory. Let C be a classification function that assigns each $e_c \in \mathcal{E}_c$ a class depending on its harshness: $C : e_c \rightarrow [0, 1, 2]$. We employ a threshold-based approach to function as C based on the sparse findings in Chapter 9. Due to the closed-source information policy on how insurance companies classify a trajectory, we

Complex
event
detection

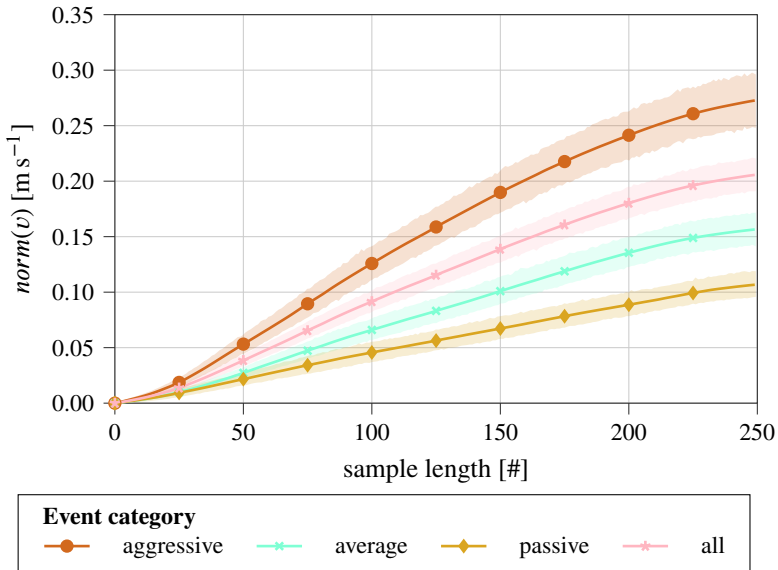


Figure 11.6 Exemplary acceleration events for a driver in the data set. Events are organized by their harshness, either aggressive, neutral, or passive which yields differences, although, events of different intensity do not overlap.

shaped C based on the available information. For example, we found that the discontinued Sparkasse premium mentioned specific numbers as thresholds. For ease of use, the corresponding classes output by C are given descriptive names, namely “*passive*”, “*neutral*”, and “*aggressive*”.

All events for a single driver are shown in Figure 11.6. It is striking that the different classes of events are distinguishable from each other. Even the span of each class does not overlap with other classes. The range of the classes, i.e. the 95% confidence intervals, show that the events’ courses do not increase significantly throughout the normalized sample. This tells us that there is a robust acceleration behavior pattern and that user acceleration events are similar. This robust and recurring behavior is a prerequisite for the events to be suitable for identification. Furthermore, on average, the depicted driver tends to drive more aggressively since the mean of the event classification is between aggressive and neutral.

Distinguishability of classes for a driver

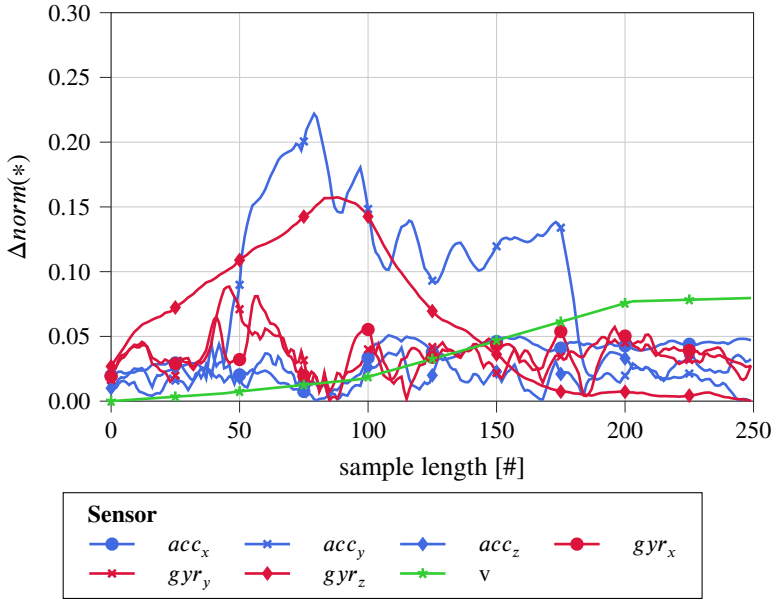


Illustration of an event across sensors. An event describes a time frame where multiple sensor readings are recorded; hence, an event can be evaluated using any of those readings.

Figure 11.7

In the context of this work in Section 5.4, events have been defined as time series that describe changes in a particular situation w.r.t. the state of a vehicle. One or more sensors in combination can define the situational change. A braking event can be described by the change in speed but also by the negative values of acc_y . Figure 11.7 shows an example event and the corresponding measured values of the different sensors, where the well-known time-based sensor fusion is applied accordingly. The figure presents the delta of normalized values. Normalization will be explained hereafter in Section 11.3.2. At this point, it is an open question as to what extent the sensors in their entirety support identification, i.e. how well they are suited to record driver-specific characteristics. This question will be discussed as part of the evaluation in Section 11.4.2. We do not transform these readings apart from normalizing and resampling them; thus, we consider them raw features.

Multidimensional sensor events

An essential aspect for the differentiation of drivers and the final identification based on events is the events' uniqueness. The better the events of the individual drivers differ, the more suitable these artifacts should be for the task. An overview

Entropy of driving events

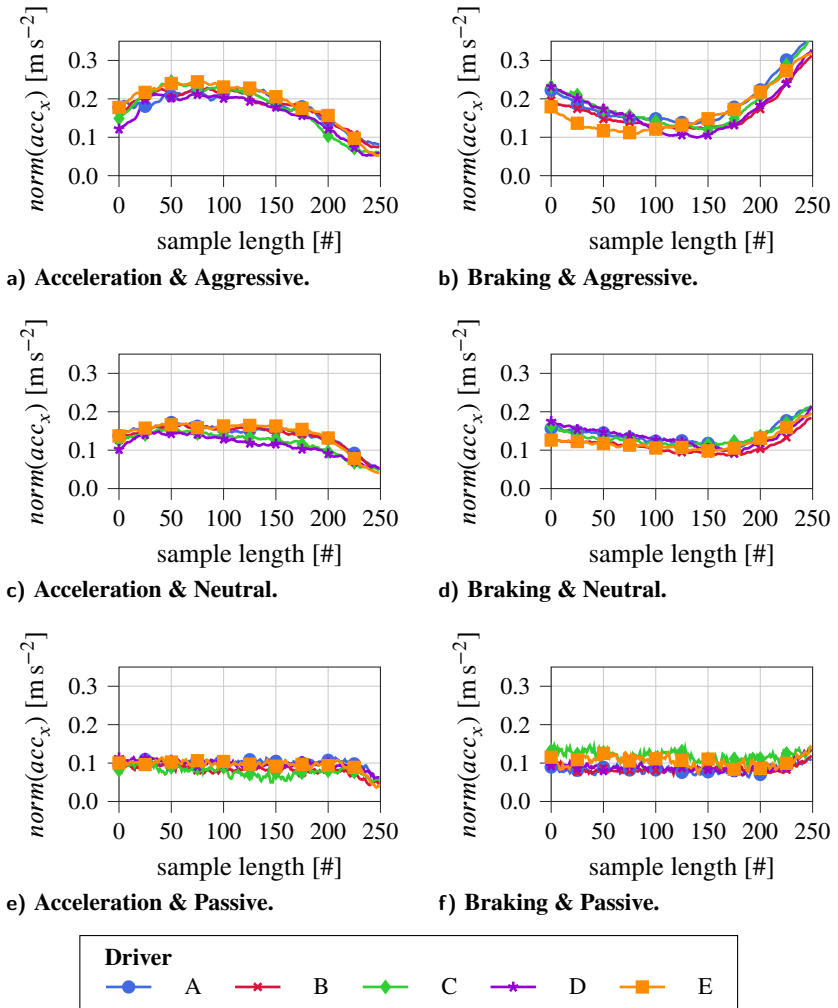


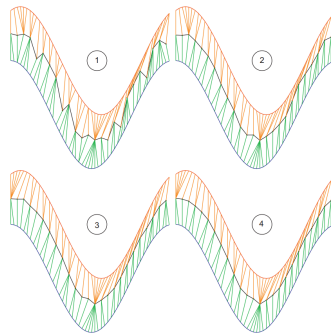
Figure 11.8 Comparison of maneuvers from different drivers and intensities. Drivers show differences in their respective driving event that eventually can be used to differentiate them. However, the difference depends on the sensor dimension used.

of the mean values of the two event classes, together with all drivers, is presented in Figure 11.8. The selected sensor is the lateral force acc_x . Regardless of event and class, differences between individual trajectories can be identified. The differences can eventually be used to assign an event to a person in the driver group. The expressiveness of a single sensor is limited. Therefore, the construction of a higher-dimensional feature vector is recommended.

Barycenter to average time series

Sidebar G

The driver identification attack in this work works based on DTW as the distance metric. There is a need to average different time series within this approach, which is challenging due to the shapes of each measurement. The keyword here is bias, as found in multiple sensor streams (c.f. Table 4.3). With *DTW barycenter averaging* a technique is described that minimizes the sum of squared DTW from the average sequence to the set of sequences [296] to eventually yield a somewhat average of the input time series. To achieve this, DTW barycenter averaging takes into account the Euclidean distances between the current average coordinate and the associated coordinates of the sequences, iteratively refining the current barycenter.



Example of the *DTW barycenter average* of multiple time series. It is computed iteratively by refining the current barycenter with updated values from each sequence. [296]

Figure 11.9

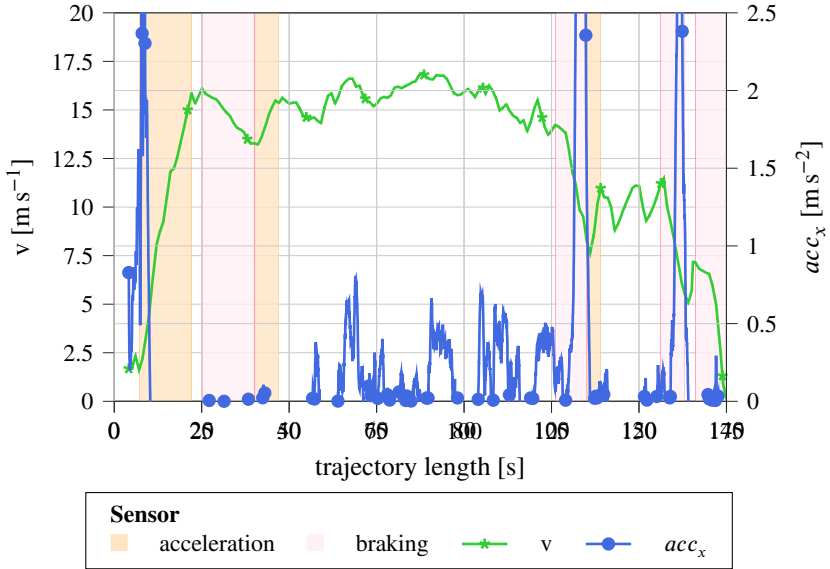


Figure 11.10 Exemplary trip with identified events. A trip is preprocessed using CEP to find events w.r.t. the given definition which are then subject to being used for identification. Areas of interest, i.e. events, are highlighted accordingly.

Figure 11.9 shows four updates to the DTW barycenter average and illustrates how the barycenter changes with each included additional sequence.

11.3.2 Identification

Identification is based on supervised Machine Learning. The task is to assign an event $e_c \in \mathcal{E}_c$ to $p \in \mathcal{P}$. \mathcal{E}_c are extracted using E_c defined in Section 5.4 using CEP techniques. Each trajectory \mathcal{M}_i yields a number of n events $[e_{c1}, \dots, e_{cn}]$ that can be used to solve the challenge. Figure 11.10 illustrates an example trajectory from a user. In total, it contains three acceleration events and three braking events, thus $|\mathcal{E}_c| = 6$. They are highlighted accordingly.

We employ k-Nearest-Neighbour classification [16]. kNN classifies an unknown object (i.e. an event) by analyzing the neighbors of that object and then selecting the most common class (i.e. a driver) among k nearest neighbors. The neighbors

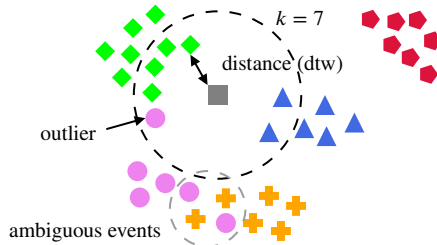


Figure 11.11

Overview of the kNN architecture. The kNN classifier uses DTW to measure the distance to each other, labeled event learned during the training phase. It then selects k classes to vote for the unknown object. Each object represents an event and each color/shape combination stands for a class. Sometimes events between classes are overlapping or out-of-bounds and therefore outliers degrading accuracy.

are selected from a set of objects for which the class is known in advance, as is the case in supervised ML environments. This classification process is similar to voting for the correct class based on the underlying assumption that similar objects are of the same class. This claim may hold based on the findings of the previous section.

kNN classification uses a customizable distance metric to identify neighborhood objects. In a simple case, this can be e.g. the size of an object. In the present scenario, however, the object to be classified is a multivariate time series in the form of an event. The class is the respective person who drove and generated the event. In the most naive case, all distances between the searched and known objects are computed, which is computationally intensive for large data sets. In the present case, Dynamic Time Warping is used as a distance metric, allowing to measure the similarity of two different time series, which in turn can serve as an input variable for the kNN algorithm. Sidebar F introduces DTW.

DTW-based distance

Figure 11.11 shows a schematic example of how the identification process is conducted. Each color/shape combination represents a driver out of the five drivers in the current pool, and each object represents an event whose class is known except for the square that is the event subject to being assigned to a class. There are five drivers in the data set that form four clusters. Two drivers seem to have a similar driving style, and therefore, some of their events are ambiguous, eventually resulting in non-discrete clusters. Furthermore, a data set may contain outliers, e.g. events that are not representative of a driver. They can

Example for determining the driver

occur at any time due to multiple effects. For example, an emergency stop may produce an unusual sharp braking pattern for a person. The drivers represented by the stars and the triangles tend to be unique compared to pluses and circles. The classifier calculates in a brute-force process the DTW distance to each of the known objects and then selects k objects where the distance is minimal. Then the class occurrences are counted, and the decision is made. In the example $k = 7$, four diameters, one circle (the outlier), and two triangles are closest to the unknown object, resulting in the classifier assuming that the unknown box must be of class diameter (and eventually the respective driver). It is evident that larger numbers of k make decisions more robust to noise within training data because the outlier does not inflict the decision. Furthermore, it is recommended that k is an odd number to prevent ties in the voting process.

Incorporating the distance metric

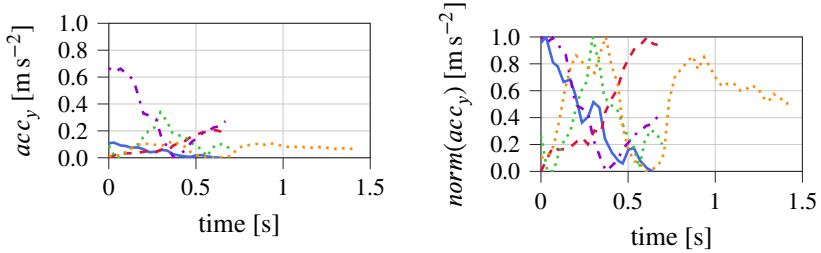
There are two options for incorporating the distance into the decision, defined by ω . First, each object in the neighborhood of an unknown element is equally weighted. This is called *uniform* distance. Second, one can use the *distance* itself to weight the neighboring object. In this case, all objects in the neighborhood are weighted by the inverse of their distance, i.e. closer objects have more influence in the voting process. We will evaluate which weighting function is preferable.

Normalization

Data originates from different smartphones and sensors that are subject to fluctuations and different value ranges (see Section 2.3). The shape of the data is in conflict with the applied ML algorithm since kNN profits from normalized data. Therefore, normalization is performed to address the problem of different value ranges. This enables comparability between multiple recordings (from various drivers, smartphones, trajectories, and vehicles) and finally makes it possible to use the classification function C.

Normalizing events

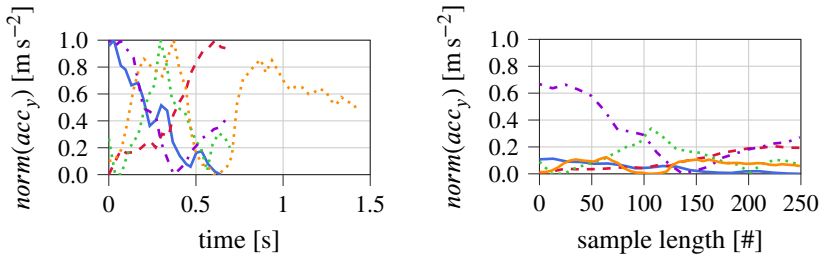
Figure 11.12 shows a comparison of the same events in a non-normalized way (c.f. Figure 11.12a) as they are recorded by a recording device (e.g. smartphone) and a normalized version (c.f. Figure 11.12b). Normalization is done by applying the maximum values seen for each user and each sensor (dimension) comprising different configurations, e.g. composed of the smartphone and the car. Experiments have shown that even though the user may have mainly passive events, some of them are still aggressive. Hence, the boundaries are likely to contain the full spectrum of values recordable for a user.



a) Unnormalized (raw) version of events. b) Normalized version of events.

Overview of one acceleration event from each driver. Sensor readings vary in range since they are recorded with different devices. Normalization can enable comparability.

Figure 11.12



a) Normalized version of events. b) Resampled version of events.

Overview of one acceleration event from each driver. Sensors readings are normalized according to maxima seen by each driver. Resampling yields time series of the same length. Resampling however does not change the course of the data.

Figure 11.13

Resample

As Figure 11.13a illustrates, the recorded events of different but also of the same drivers are most likely of various lengths. This limits the comparability, even if DTW can, in principle, handle such effects. Crucial for detection is the shape and the corresponding similarity of two events. For example, it is of interest whether a driver brakes homogeneously or draws a more divergent picture during such an event. A braking event with a delta of several 10 km h^{-1} over a long period is comparable to an event that has a smaller delta but is also shorter.

Uneven event lengths

Resampling can be used to bring the events to equal length without changing the shape of the data series. It is producing a new discrete time series based on the underlying signal of the continuous time series [286]. This is explicitly shown in Figure 11.13b. If some events were exceptionally short and had a high

Resampling to enable comparability

delta, the drop in values is put into perspective when the data series is resampled. According to DTW there is no change in the measurement points because all time series have the same length and value w.r.t. the sensor values.

Note on the implementation

We employ the *TimeSeriesResampler*² of tslearn, a Python library for Machine Learning time series data. Resampling is the change in the sampling rate of the incoming data. Let us say that the incoming data is collected with $f = 25$ Hz in our case. We now interpolate the underlying signal using a specific number of points from the continuously incoming data called ρ and connect them accordingly. The relationship between the sampling rate and the underlying phenomenon is illustrated in Chapter 2. The result is a trade-off between the complexity of the resampling task and the accuracy with which the original signal is mapped. It must be ensured that ρ is large enough not to lose the shape of the initial data, ultimately ensuring identification success.

11.3.3 Scoring

The uniqueness of an event is incorporated into a score that represents the confidence and probability that the classifier assigns a series of unknown events to a person in the pool \mathcal{P} . We assume that a driver does not change during a trip without stopping. Hence, all n events must be generated by the same p : $[e_{c1}, \dots, e_{cn}] \rightarrow p$.

Table 11.3 Overview of a trip evaluation to eventually predict a driver. The trip contains five events of the same type with five potential drivers. The prediction is based on a driver's likelihood to be the correct driver

Driver	Event					Pred	Pred
	1	2	3	4	5		
A	0.00	0.00	0.00	0.07	0.07	-0.87	-1.00
B	0.53	0.13	0.60	0.27	0.13	0.67	1.00
C	0.27	0.13	0.13	0.40	0.20	0.13	0.30
D	0.07	0.40	0.07	0.07	0.33	-0.07	0.04
E	0.13	0.33	0.20	0.20	0.27	0.13	0.30

²https://tslearn.readthedocs.io/en/stable/gen_modules/preprocessing/tslearn.preprocessing.TimeSeriesResampler.html

Table 11.3 presents, using an example with $n = 5$ events, the scoring process. First, the algorithm assigns each event e_{c1}, \dots, e_{cn} individually to the drivers with a certain probability. The higher the value between 0 and 1, the more certain the algorithm is. In the example, the probability that driver B created e_{c1} is 57%. In contrast, e_{c2} is assumed to have been driven by driver D. e_{c5} is more challenging to interpret because the probabilities are close here.

Example on determination

A simple frequency analysis would under-measure the probability in the prediction and discard valuable information. Both driver B and driver D would have two votes in the present case. Therefore, a score is calculated per driver for all events of the trip. The calculation is done as follows. On average, the probability that a driver produced an event equals $\frac{1}{|P|}$. The deviation from this value, positive as well as negative, thus qualifies the statement of the classifier. For example, values of 0.25 with an average probability of 0.2 are not very meaningful because they are close to the guessing probability. Such estimates should be considered less in decision-making than, for example, e_{c1} and e_{c3} , since here, driver B was chosen with a high probability. The summed normalized probability results in a total score denoted “*Pred*”. The range is subsequently normalized to a uniform range of values between $[-1, 1]$ (“*Pred (norm)*”). The decision is based on this value. In this case, driver B is the predicted driver, corresponding to reality.

Identification of the optimum of the prediction

Decision

11.3.4

The decision is made on the events found in a trajectory. However, we introduced different types of events. Depending on e.g. the trajectory, external circumstances, and vehicles, some events are less informative than others. We also found that two drivers may produce similar events, although their general driving styles differ. For example, in stop-and-go scenarios, braking events from multiple drivers may look similar due to external circumstances. In principle, unique trips or events should be preferred since they allow for a correct statement. Therefore, for each trip, a *confidence* is calculated, which corresponds to the standard distribution of the prediction (c.f. Table 11.3 *Pred* column). For example, this results in $\sigma = 0.56$. The higher this unbound value, the more unique the trip.

Putting events into perspective

We call this *maximum-based selection* in contrast to *average-based selection*. The average-based selection model would choose the driver who was predicted the most across event types, even though some events are less meaningful because of mentioned reasons. It eventually results in a worse prediction.

Maximum-based selection and average-based selection

11.4 Evaluation

We now present an evaluation of our algorithm according to our experimental setup. First, an analysis of sensor significance to construct the feature vector is presented. Next, the actual identification rate is discussed based on alternating driver configurations. Finally, the learning rate, i.e. the number of events required to draw meaningful conclusions, is discussed.

11.4.1 Preamble

We extracted all the events of each driver and created a pool of events of the same type (i.e. acceleration and braking). Then, that pool was shuffled. From that pool, we selected 80 % training data and 20 % test data. Furthermore, the training data was split again using an 80-20 split to provide a validation data set. We applied k-fold cross-validation with $k_{fold} = 4$ to prevent overfitting when learning at all experiments. Consequently, the algorithm's accuracy was assessed using test data that only included events that it had never seen before. They were either given to the identification algorithm separately or in the form of an artificial track. Artificial tracks are generated by concatenating a given number of randomly selected events from a specific driver.

Table 11.4 Overview of hyperparameter options and selected values for the identification approach.

Parameter	Value
k	[3, 5, 10, 15 , 25, 50]
ω	[uniform , 'distance']
η	[dtw , 'softdtw']
ρ	[10, 25, 50, 100, 150, 200, 250 , 500]

Hyperparameter optimization

The presented approach for the identification of drivers based on events is based on ML. The approach uses DTW as a metric for the kNN algorithm. We optimized respective hyperparameter w.r.t. accuracy. Table 11.4 shows all parameters that were optimized using a scikit-learn grid search. Furthermore, the 65-15-20 split with k-fold cross-validation was used. We found that larger values of k are preferable; however, runtime increases. Hence, a trade-off was made. The same holds for the resampling rate ρ set to 250, which also provided a good balance between accuracy and runtime.

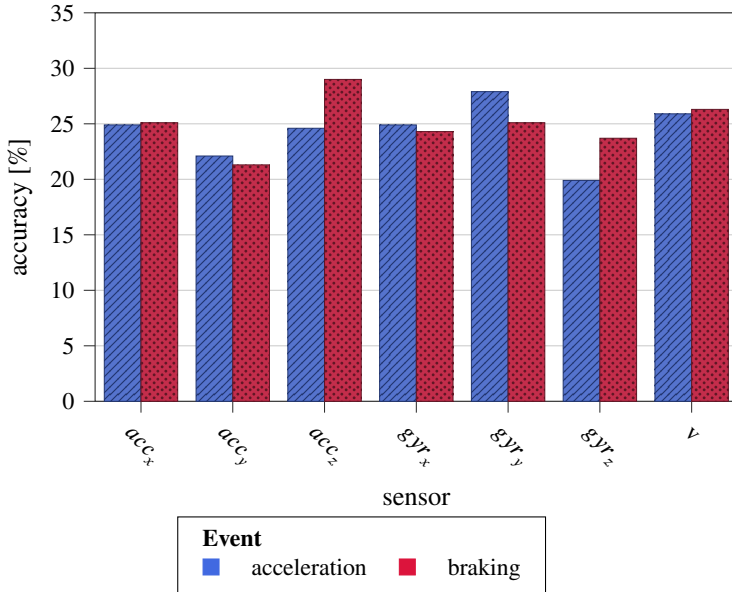


Figure 11.14

Overview of the achievable accuracy for assigning an event to a driver. Using one-dimensional feature vectors only containing readings from one sensors for a data set of five driver yield results above the guessing rate, although, no sensors is favorable for standalone use.

The experiments for each analysis were performed at least three times with a set of unique random seeds to guarantee different slices of data subsets. This should result in a more generalizable output and reduce the dependence on event selection.

Repeated execution of experiments

Relevance of Features

11.4.2

As mentioned in Section 11.3.1, an event is a collection of sensor values for the time interval of that event. The data of the multivariate time series includes the accelerometer acc , gyroscope gyr and the derived speed v . This data ultimately results in a feature vector of \mathbb{R}^7 . The feature vectors should be small in dimension and not include unnecessary information to ensure effectiveness. Therefore, we analyze the accuracy achievable with a specific sensor by creating a one-dimensional feature vector of \mathbb{R}^1 for the DTW-kNN-based approach.

Successively reduced feature vector

Balanced
feature
relevance

Figure 11.14 shows the accuracy that can be achieved when assigning an event to a driver from the driver pool. The data set includes all five drivers. The values are then averaged scores from each test and validation data set. It is trivial to record that no sensor is favorable for standalone use, but also, no sensor performs significantly poorly in terms of accuracy. Consequently, all sensors should be used for further processing. Furthermore, any combination of one-dimensional feature vectors yields a result above the guess rate, apart from the fact that some sensor types are less meaningful for a specific maneuver than others. It should be noted that the significant sensor types for acceleration and braking differ, although one maneuver is the opposite of the other.

11.4.3 Impact of Driver Set Size

Uniqueness
of drivers

The identification algorithm uses previously learned events of drivers to assign unknown events to the known set of drivers (closed-set). The presented approach can use two different strategies to use events for the assignment task, but the select-best approach is used for the evaluation in this scenario. The literature has shown that the task of identifying drivers becomes harder the more drivers are in the group (e.g. [190]). This hypothesis also holds in our case. As Figure 11.15 illustrates that more events on a trajectory support the identification process, so the accuracy of up to 96.4% is achieved for 14 events for a data set size of five drivers. The high accuracy can also be maintained for other data set sizes. However, the difference is that the identification rate quickly increases. Interestingly, the accuracy varies for data sets of two drivers and reaches lower levels. Analyzing this fact results in dependence on the formation of driver combinations. The approach uses DTW as a distance metric that measures the distance between two events. Lower distances imply that some two events are similar. Simplified, they could therefore originate from the same person. Therefore, it is reasonable that identification accuracy decreases with more drivers in the data set because the likelihood of similar driving behaviors among drivers in the set increases. The kNN-based approach may select the wrong individual from its knowledge pool.

Acceleration
in favor of
braking

Breaking down the identification to an event level from the trajectory perspective allows understanding of the significance of event types. The current implementation uses acceleration and braking events. Figure 11.16 presents the boxplots categorized into set sizes and event types. First, the identification accuracy for the driver pool decreases with more participants. Apparently, acceleration events are better suited for identification as such events yield higher rates while keeping at the same time smaller ranges. For example, the lower whiskers of braking

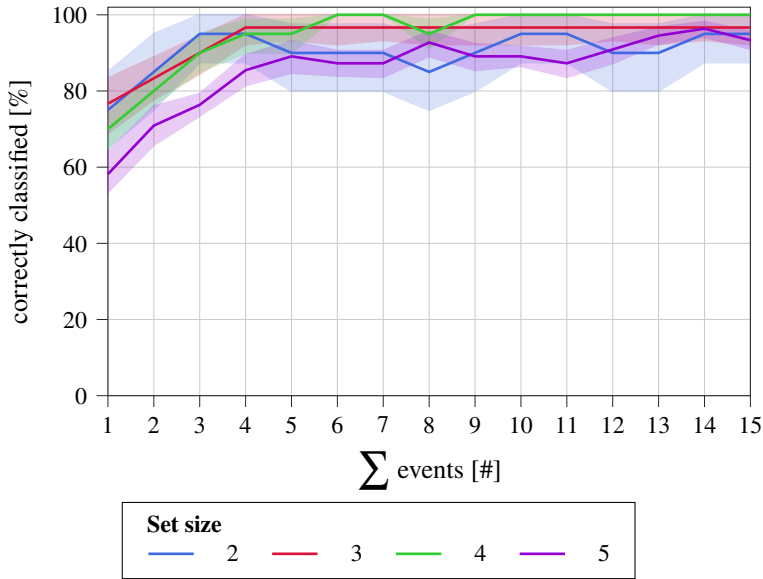


Figure 11.15

Overview of the number of correctly identified trips as a function of test size and number of events. Larger test sets require more events to be present in a trajectory to yield high identification rates.

events are, with 45 % for two drivers, significantly lower and wider than acceleration events. This trend holds for larger set sizes. In conclusion, acceleration events are favorable for identification because they are more reliable and robust for the task. In particular, all rates are above the chance, even for outliers.

From a driver-focused perspective, one can see in Figure 11.17, that the confidence of correctly identifying a trip decreases as expected with larger set sizes. This does not necessarily reflect a decrease in identification accuracy on a trip level. The experiments were carried out on trips with 15 events. Multiple trips per driver were tested, whose confidence is represented by error bars with 50 % confidence interval. Depending on the events found in a trip, the trip identification confidence varies larger for smaller data sets. There are differences between drivers in how they can be identified and how unique their driving maneuvers (events) are. Driver E is consistently conspicuous and thus easy to identify. On the other hand, driver A does not appear to depict a significant driving pattern. It also becomes more challenging to identify driver B with larger data sets. This driver seems to be relatively unique in his driving behavior. This driver seems

Unequal driver accuracy

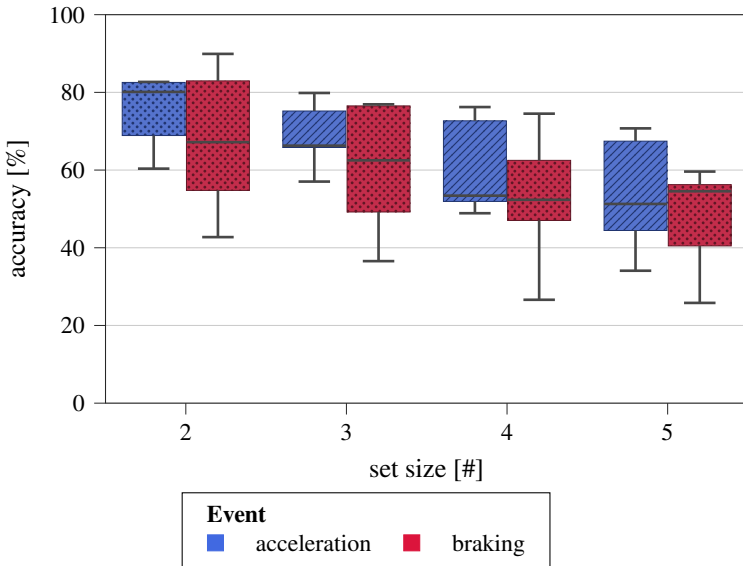


Figure 11.16 Accuracy for detecting a single event depends on the set size. Set sizes that include more drivers degrade detection performance. Acceleration events are easier to assign to the correct driver in comparison to braking events.

to be relatively recurrent in his driving behavior but not unique. The varying confidence of drivers is also reflected when taking a closer look at the confusion. The confusion matrix in Table 11.5 presents the assignments for each driver, each having made twelve trips with 15 events of each kind. Trips of drivers A, C, D, and E were correctly classified in each case, even though the algorithm could not assign each event correctly. However, the maximum-based approach works well in the given scenario, compensating for minor uncertainties. The results align with those discussed above that yield driver B as a driver without a unique driving pattern.

Summary Together, the identification approach based on DTW and kNN seems feasible for the task of identifying driver events. It presents high accurate identification rates above the guessing rate, even on an event-focused level. Furthermore, our proposed approach can successfully identify a driver with longer trajectories containing more events, likely for urban trips. Recall that a trip contains 58 acceleration events and 57 braking events on average for the data set, with a minimum of four and one, respectively.

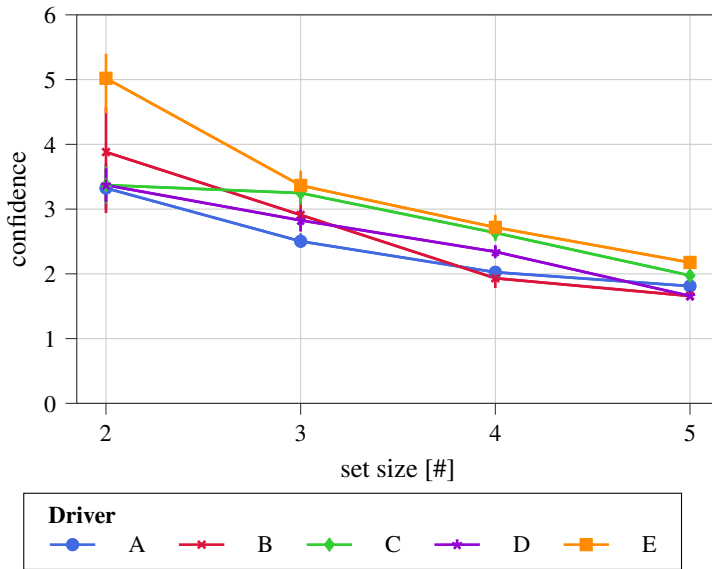


Figure 11.17

Accuracy for correctly identifying a the driver of a trip. Acceleration events are easier to assign to the correct driver compared to braking events. The 50 % confidence interval shows larger fluctuations in confidence for smaller set sizes.

Confusion matrix for driver identification after twelve trips each. Each trip has 15 acceleration and 15 braking events of different harshness. Trips are normalized based on the true label (rows, i.e. twelve trips).

Table 11.5

		Predicted				
		A	B	C	D	E
Actual	A	100	0	0	0	0
	B	0	50	8	0	42
	C	0	0	100	0	0
	D	0	0	0	100	0
	E	0	0	0	0	100

rates in percent

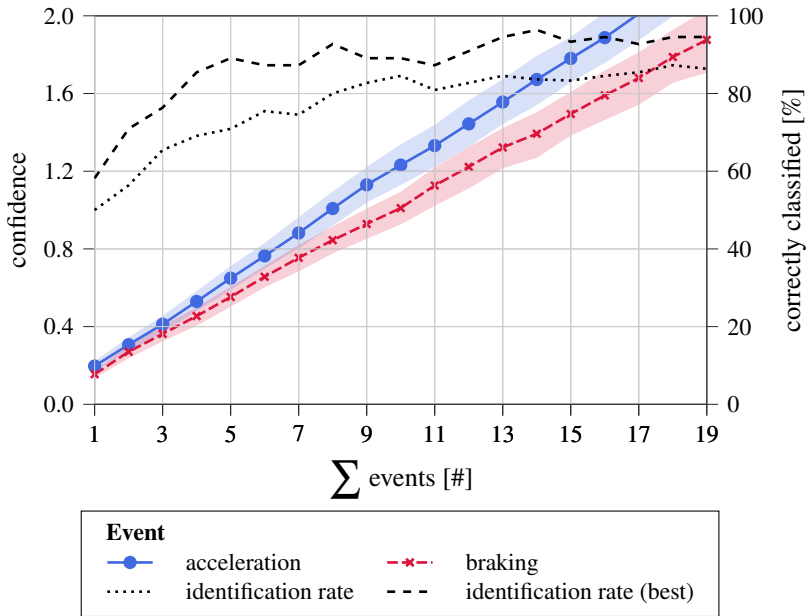


Figure 11.18 The confidence of a identification guess increases with longer trajectories providing more events eventually analyzed. Acceleration events tend to provide more information usable for identification. Longer trajectories yield high identification rates.

11.4.4 Event significance

Detection is subject to multiple factors

The amount of information yielded by an event may differ depending on the event's progress itself. External factors such as traffic also influence the driver's driving behavior. For example, in stop-and-go traffic scenarios, braking events of multiple drivers can be similar to a specific degree. This makes it harder for our approach to differentiate the drivers since the distances between (or similarity of) events that work as an indicator are lower (or higher).

Correlation between number of events and confidence

However, the PHYD scenario allows us to conclude the identity of a driver based on a recorded trajectory. Therefore, we analyze the impact of the number of events, such as the recording, on the confidence in the identification. The confidence is higher if the algorithm is more sure of guessing the correct driver. For this experiment, we simulated ten different trips for the full data set of the five drivers. The results are aggregated using a mean with the interval range included in Figure 11.18. The figure plots the confidence against the number of

events. It is evident that a higher number of events in a trajectory results in a more confident decision. Values above one allow good decisions. The figure also shows that there is a difference between the event types. Acceleration events are superior to braking events, i.e. they tend to represent a driver's driving behavior more fittingly. This seems logical since braking events are often induced by external factors such as the next vehicle that changes its speed. This is different for acceleration events. One can see that acceleration crosses the confidence if seven acceleration events are found within a trip. On the other hand, braking requires at least ten events for the same confidence level.

The presented identification approach uses a select-best strategy that considers the previous finding. It prefers events with higher confidence. For example, if a trajectory contains two brake events and two acceleration events, one of them has higher confidence in each case. In this case, it is not meant to balance all four events but only to use the most significant ones. The difference between the select-all and select-best approach is also shown in Figure 11.18. The dotted line is the select-all approach, which yields a lower proportion of correctly identified drivers. The select-best method outputs identification accuracies of 100 % for a low of only five events of each type found in the trajectory.

Select-best
strategy

Learning rate

11.4.5

Previous experiments were conducted using the whole data set split into the proportions mentioned. To assess the learning rate of the algorithm, the training and validation sets were reduced iteratively in steps of 10 %. The test data set was not adjusted. Events are randomly selected; hence, the number of events per driver in the data set is not guaranteed. In the worst case, no events from a driver might be in the reduced data set. However, we reran the experiments three times with different seeds to reduce random effects.

The learning rate shows an interesting pattern, and it is expected to increase with more data available for training. This is common in other ML tasks. However, the path here is different but still has a linearly increasing shape, as Figure 11.19 shows. This indicates that the algorithm is learning more the more events of a driver are seen.

Resistance-
led
development

For the full data set example shown in Figure 11.19a, the accuracy for both events starts at 20 % accuracy, which at the same time equals a guess (more precisely, this only holds if each driver is equally represented in the data set, but for the sake of simplicity, we stay with this facilitation). At 30 % of available training data, the identification accuracy based on acceleration and braking

Full set with
all drivers

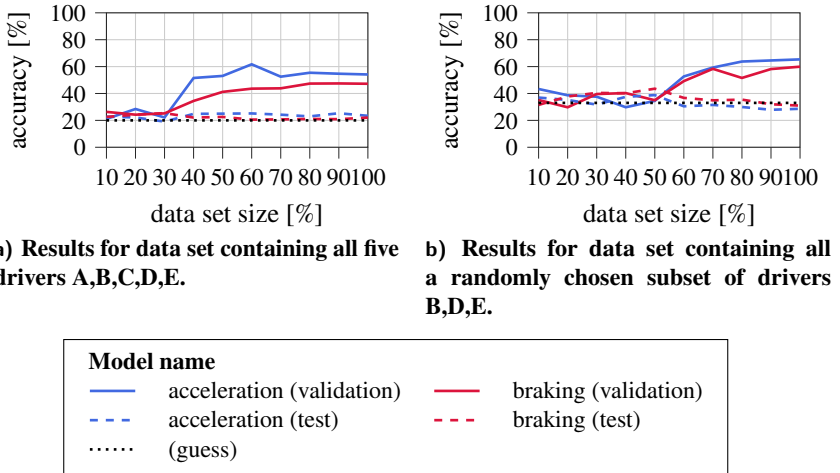


Figure 11.19 Overview of the learning rate of the algorithm with different amounts of training data available. The accuracy increases the more data is used to train the kNN model.

events significantly increases until it reaches the global peak around 70%. While the rate of identification of braking events increases monotonically, the acceleration curve has a peak at 60% and then decreases around 5%. There may be two assumptions as to why this happens. On the one hand, the algorithm might experience significantly different events from those seen before. Consequently, the model is also adjusted to fit the new data; however, the vagueness increases. On the other hand, this effect may be related to the construction of the data subsets and the intended over- or underrepresentation of a driver in the data pool.

Validation vs.
test set

The accuracy within the test set is lower than those for the validation set. This may imply that the model is underfitting. Concludingly, more events by a person are needed to label events accordingly. However, in a trajectory-based view with more events (c.f. Figure 11.15), a high accuracy of single event identification is not needed. This stresses the feasibility of side-channel attacks on sensor data and educated guessing with domain knowledge. At the same time, the presented approach is also proof of the imminent privacy threat (see Chapter 10). If the size of the data set is 100%, the acceleration and braking validation score shows the same accuracies as seen in Figure 11.16 for a data set size of five drivers.

The task of driver identification is simpler for smaller sets of drivers, as discussed earlier. This claim holds for the example shown in Figure 11.19b. For a small slice of the data set, it seems that newly observed event shapes are introduced. The algorithm struggles to detect them but ultimately increases its accuracy in the given task.

Learning rate
for different
set sizes

Realistic Environment

11.5

The presented attack shows that an insurer in the UBI insurance environment can trace which person drove a route based on the transmitted data. Thus, the outlined threat scenario for the user's privacy can be confirmed. The purposefulness of the data cannot be ensured, and, ultimately, the data submitted is too broad.

The previous attack is based on two crucial assumptions that are reasonable in the given context.

Obsolete
assumptions

- ▶ First, the number of drivers per vehicle must be known so that an insurer can create a data pool \mathcal{P} accordingly. Consequently, an identifier (e.g. p) must be defined for each unique driver. Identifiers must not be shared between different drivers and are used exclusively by that driver (**static driver set**).
- ▶ Additionally, the data evaluator must have information about which driver was driving at any given time. He is then able to match the driving distance to a driver from the data pool: $\mathcal{M} \rightarrow \mathcal{P}$. Drivers are not allowed to switch during a measurement \mathcal{M} (**labeled trajectory set**).

Therefore, based on assignment (\mathcal{M}, p) , the labeled data set required in supervised learning is created, on the basis of which the kNN-DTW algorithm can be trained. As in Section 11.4, the algorithm is then able to assign a trip \mathcal{M} to a person p from the driver pool \mathcal{P} with high probability, without the need to transmit an identifier. Unprocessed sensor data or the subset containing all events $\mathcal{E}_c \in \mathcal{M}$ is sufficient.

The proposal represents a side-channel attack in the context of sensor data and can be categorized as a fingerprinting attack. The goal is to detect and possibly track a user regardless of short- or long-term identifiers (c.f. Section 10.3). However, the assumptions made may be overstated for the PHYD scenario. For example, it can be questioned whether the second assumption that a route is

Reflection on
the PHYD
scenario

already known in advance to an attacker may hold. The ML problem thus shifts from a supervised learning problem to an unsupervised learning problem, where the target class is not known *a priori*. This is considered a clustering task instead of a classification task. Nevertheless, the threat to privacy still exists if an attacker can correctly build clusters of drivers without specifically naming them. We also challenge the *static driver set* assumption by introducing a new driver to the pool where no or only a few trajectories are known. This yields a severe underrepresentation of the respective driver.

11.5.1 Moving to a Clustering Problem

Replaced
classifier

We now loosen assumption two. The pipeline is adapted accordingly for the following test series. All steps of the preprocessing up to normalization and resampling remain the same. Only the ML algorithm is replaced. Instead of a supervised learner *KNeighborsTimeSeriesClassifier* an unsupervised clustering algorithm called *TimeSeriesKMeans*³ is used to identify drivers with the distance metric kept the same (i.e. DTW). We reran the experiments with the altered environment. They present a silhouette coefficient of -0.02 over all clusters as shown in Figure 11.20b. The silhouette score allows interpreting the performance of a clustering approach by giving insight into how an object is assigned to its cluster compared to being separated from other clusters [328]. The values can range between $[-1, 1]$. Perfectly assigned and separated clusters have a value of 1, while confusing and overlapping clusters with neighbors yield a score of -1 .

Limitation of
the driver

Given that, the scores around zero imply that the clusters overlap. The events between drivers are likely similar, yet they are usable for identification. Compared to a two-driver scenario as illustrated in Figure 11.20a, confusion and non-separability between clusters increases significantly. An insurer might not make a clear decision on this basis, but it still allows them to narrow down the options in terms of possible drivers because it may be unlikely that events always overlap with the same cluster combinations. It should also be noted that this score refers to an independent consideration of individual events. In Section 11.4.3 it has already been shown that the maximum-based selection process provides significantly better results in the present scenario and eventually allows the correct identification in most cases. It stresses the fact that domain knowledge is essential.

³https://tslearn.readthedocs.io/en/stable/gen_modules/clustering/tslearn.clustering.TimeSeriesKMeans.html

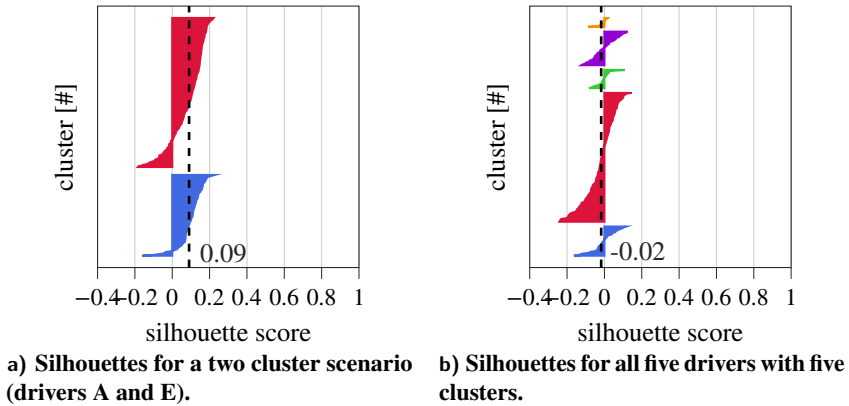


Figure 11.20

Silhouettes scores for a different number of clusters. The number of clusters equals the number of drivers in the data set. The silhouettes coefficient is roughly zero for most scenarios indicating no clear separation between drivers.

Introducing New Driver

11.5.2

Within the evaluation, the impact of the data set size, i.e. $|\mathcal{P}|$, was analyzed. The more drivers there are, the more complex the identification task becomes. However, the impact of the number of events was not subject to analysis. Thus, we introduce a new driver $p' \rightarrow \mathcal{P}$ that is underrepresented in terms of the number of events. Fewer events may not give a comprehensive impression of a person's driving style, which is crucial for identification based on DTW.

The chosen *KNeighborsTimeSeriesClassifier* provided by *tsfresh* does not support open-set scenarios. Introducing unknown drivers to a data set results in shifting the closed-set problem to an open-set environment. In order to assess the implications in quality, we added a dummy event for the unknown driver to the data set that cannot be used for learning due to its shape. Our approach cannot detect the new driver but confuses other drivers. In particular, confusion does not occur with the exact driver for each of the ten different trips that we created for the unknown driver. The identification results are not surprising when an unknown driver is present in the data set, and this is somewhat similar to one driver being underrepresented to the most extent. Therefore, the algorithm cannot derive a typical driving behavior for that driver. Furthermore, an unknown driver also degrades the accuracy of all drivers. Another driver was not correctly identified on two out of ten trips, which was no challenge for the clean data set that excluded the new driver.

Simulating unknown drivers

11.5.3 Non-uniqueness of Drivers

Now we analyze how discarding both assumptions affects the identification. For example, it is assumed that two users use the same smartphone, which is used to transmit data to the PHYD insurer. Whether this is a breach of contract on the part of the drivers is not to be judged here. A trip tuple can then look like $(\mathcal{M}, p_i \vee p_j)$ with $i \neq j$.

Shuffling trajectories and drivers

The following approach was chosen to measure the impact of sharing the data management devices or the respective data submission channels.

1. Select a subset of drivers $\mathcal{P}_M \subset \mathcal{P}$
2. For each driver $p_l \in \mathcal{P}_M$, select a fraction $frac = 0.5$ trajectories $\mathcal{M}_{l,i}, \mathcal{M}_{l,i+1}, \dots$ from his pool of trips, i.e. 50 %.
3. Change label of such trips (\mathcal{M}, p_x) so that the previous driver association is altered $(\mathcal{M}, p_{x'})$ where $x \neq x'$ ($x, x' \in \mathcal{P}_M$)

This yields trajectories in which either one driver may use his device or accesses another one to submit data in the UBI context. In contrast to a sound data pool used for training, accuracy decreases to 41.7 % (from 51.5 %) for acceleration events and to 38.3 % (from 43.5 %) for braking events, respectively.

11.6 Conclusion

In this chapter, we have underlined the threat posed by the transmission of raw data but also aggregated information in the form of events in the PHYD business model. Several user claims about privacy can be violated.

11.6.1 Overview of Literature

Non-overlapping environments

First, a SLR was used to find methods that allow the identification of a driver based on sensor data. Most approaches have been shown to use vehicle data collected through well-known interfaces such as OBD. These approaches conflict with the PHYD scenario. Here, most of the vendors use smartphone-based applications and IMU data. There are also apparent differences in the features used. Although the analysis of the UBI premiums showed that acceleration and braking behavior, as well as steering actions, play a role in the pricing process, more specific features can also be found in the approaches from the literature. Many rely on vehicle data such as pedal operations or engine characteristics. These

do not lend themselves to serving as features for an identification attack based on smartphones. Here, the limitations that are described in Chapter 2 become apparent in terms of the variety of data that can be collected. With acceleration and braking, known events can be found again. These were consequently used for our approach.

No standard approach or recommendation appears after analyzing the literature. Due to the heterogeneous data sets, no comparison of the performance of the approaches from the literature is possible. The type of data set also differs. Only a few works use real-world data with random trips. However, random trips are the default case in the PHYD scenario. Thus, an identification attack must be able to deal with them. Furthermore, recorded trajectories cannot be assumed to follow specific patterns (e.g. a corner driven by each person in the data set). A robust identification approach has to be designed to take that into account. All works see the task of identification as a supervised learning problem, which becomes more difficult as the size of the driver pool increases.

No consensus on identification approaches

Our Approach

11.6.2

Based on the insights from the literature, we decided to apply distance-based learning to the problem of the identification of drivers. To the best of our knowledge, we are the first ones to use DTW in combination with kNN classification. Our approach is based on events extracted from the literature and introduced in Section 5.4, namely acceleration and braking. The assumption is that similar events, i.e. events with a small distance, are from the same driver. The attack only uses data collected in the PHYD environment and can be further transferred to use zero-permission sensors solely.

Event-focused approach

Our proposed pipeline respects the heterogeneous origin of trajectories and individual driving behavior. It uses CEP to extract relevant data from the whole trajectory. It is assumed that one driver generates one trajectory without switching. Then it applies normalization to each event. Normalization boundaries are driver-specific and are guessed by using a min-max greedy search on all events. The events are then resampled to a specific length. Normalization and resampling ensure that the shape and progression of events are not altered and that the underlying assumption still holds.

Process of the approach

We found that data from the accelerometer and the gyroscope along all three axes are equally important, and velocity is also included in the feature vector. Experiments show that the accuracy of assigning an event to a driver is average. However, domain knowledge allows drawing further conclusions from the

Result summary

classifier's prediction: Each trip usually contains multiple events assigned to a driver. This can support the decision within a scoring and selection process called maximum-based selection. Using maximum-based selection, we found that a trajectory with five events is sufficient to correctly assign each trip to a driver within a driver pool of five in more than nine out of ten runs. The number of events needed is decreasing with smaller driver groups. That finding corresponds with what we found in the SLR.

11.6.3 Privacy Implications

As introduced previously in Section 1.2, Solove [351] considers privacy essential for protecting multiple rights of a user. As we will see, the identification approach in the context of PHYD violates some of these rights. Some of them are discussed now. Of course, some rights are excluded or less important due to the context.

Limit on
power

First, privacy should enable *limit on power* that is degraded by sending raw sensor data. A counterpart may also use the data for other purposes, as shown. This aspect is closely related to the purpose limitation required by the GDPR. Purpose limitation is no longer given as soon as a data processor can also extract other information from the data than is necessary for the actual business model. In the case of UBI, this is the price of a vehicle based on how it is driven. The driving behavior to which the driver contributes in total is, by definition, irrelevant. In Section 9.2, this aspect was addressed in a comprehensive way. At this point, it can be argued that providers already practice data minimization by working exclusively on aggregated data, i.e., events. However, the aggregation does not necessarily occur in the user's domain but by a third party, which delivers aggregated information to the insurer. This approach is insufficient. In Chapter 14 a procedure is presented that can achieve data minimization by evaluating the data locally, and accordingly, only the business model can be performed based on the data. Identification is then no longer possible.

Trust

Solove [351] also argues that *trust* is an important right of the consumer. Trust can only be established if a user feels adequately secure and autonomous. He must assume that his data will be used exclusively per the other party's reasoning. Due to bounded rationality, it is also necessary for the user to ensure that the disclosed information is handled securely. He is not in a position to check the insurer's behavior and must trust that this will be done by other bodies (including the insurer) themselves. His privacy should not be vulnerable if he follows the specified processes. Unfortunately, this is still the case.

The rights of *reputation management* or *control over one's life* must also be viewed critically. If the pricing process is broken down from the vehicle to the individual driver, this enables effective tracking of routes and extraction of points of interest from people. It is thus possible to project locations onto individuals' daily routines and create detailed movement profiles that are used as domain knowledge in further steps. In terms of reputation management, an individual should be allowed to move anonymously without revealing specific locations such as hospitals or workplaces [132]. An individual's reputation may suffer depending on where they are. In order to pinpoint a user's location, additional location-inference attacks based on zero-permission can be employed. We will present such an attack in the next chapter.

This list is just an excerpt of how the misappropriation of data violates a user's privacy. Of course, dangers can also be applied to a broader context outside of the UBI scenario. In terms of a side-channel attack, the permissionless sensors of IMU can also be wholly intercepted without the user's knowledge.

Reconstructing Trajectories Based on Sensor Data

12

After the previous chapter presented a fingerprinting attack to identify a driver based on sensor data, this chapter looks at another class of side-channel attacks, namely location inference. It is the second part of our attack tuple. This type of attack attempts to use sensor data to determine a target point or a person's entire travel distance and is intended to demonstrate the threats to location-based privacy.

With this attack, we recall the problem of the information disclosure process based on cognitive perception as we introduced it in Section 1.2.3. Moreover, the attack is settled in the UBI environment and addresses i.a. the topic of Nothing-to-Hide [350] to stress the fact that aggregated data can provide a comprehensive insight into a person's daily life, ultimately proving that this bias is a false friend. Also, hyperbolic discounting is addressed as the short-term benefit of UBI may be a monetary reward, but long-term knowing the daily schedule allows one to derive information such as sickness (when visiting a hospital).

Reoccurring
challenges

Even though multiple methods for trajectory inference exist, with each having individual restrictions, we propose an evolutionary approach that explicitly addresses the said drawbacks. Therefore, we contribute with

Contribution

- ▶ a survey on methods for trajectory reconstruction to understand the respective circumstances,
- ▶ a powerful yet efficient attack to derive a trajectory for large areas based on zero-permission sensor data, an

Acknowledgement

Parts of the research presented in this chapter are based on work published previously [324] and supervised work [S7].

- ▶ a comprehensive evaluation of the performance of our proposal, including a comparison against state-of-the-art related work based on multiple real-world data sets.

The evaluation in this chapter is i.a. based on data sets R1 and R2.

Structure We first perform a survey on recent approaches for trajectory inference in Section 12.1 and present the results in Section 12.2. Based on the knowledge generated, Section 12.3 introduces an evolutionary method for the reconstruction of the route, including the fundamentals needed to understand the given scenario. Then, a presentation of the building blocks required to perform the task is explained in Section 12.4. An in-depth design explanation of the attack follows (see Section 12.5). Based on real-world data, a thorough evaluation in Section 12.6 assesses the performance of our trajectory reconstruction approach. We conclude this chapter in Section 12.7.

12.1 Structured Literature Review

To identify related work, we performed a SLR whose methodology we present in this section. Based on the research questions, the search process is defined. The relevant results are then listed and quantitatively described.

12.1.1 Research Question

The literature review aims to provide an overview of route tracking methods. The condition is that these are applicable as a side-channel attack based on smartphones. Therefore, the main research question is “*What is the state-of-the-art of route reconstruction approaches enabled by smartphones?*”. The question is divided into the following questions to investigate the respective components.

- Q1** What are preconditions for reconstructing a route assumed or significantly required? What sensors will be used?
- Q2** What different methods and approaches are used for trajectory inference? Are there existing standard procedures, if any?

The research question will be answered subsequently in the next section by analyzing the respective literature. This section includes work by Heiland [S7] that performed a SLR on trajectory inference.

Search Process

12.1.2

The search process for the SLR is based on Kitchenham and Charters [209].

We selected ACM Digital Library, SpringerLink, IEEE Xplore, and ScienceDirect as libraries to look for relevant research. This study focuses on approaches that can be executed with the support of a smartphone, consequently excluding works that are not feasible given this constraint. Therefore, the following search string¹, based on the variable for the approach and one for the recording device, was applied to all libraries.

Search term

```
( MAP MATCHING | [ ( ROUTE | PATH | TRACK ) ] )
& [ ( TRACKING | ESTIMATION ) ]
& [ ( SMARTPHONE | MOBILE DEVICE) ]
```

The search term “mobile device” may also include devices such as smartwatches; however, we removed such work in a later process. We found that these works are not compatible with the given scenario of PHYD due to the different shapes of data that are acquired. For example, a smartwatch may be equipped with an accelerometer, thus collecting similar data to a smartphone, yet the validity is not comparable. No year constraints were presented, although the environment with smartphones or mobile devices naturally yielded a young research field. We allowed ourselves to include work that seemed relevant even though it had not been identified using the given procedure (i.e. forward search and backward search). As previously, duplicates are eliminated.

In particular, we developed the following inclusion and exclusion criteria to identify relevant work. Articles that met the requirements were analyzed in-depth with a full-text read to answer the research questions.

Inclusion
criteria

1. The approach tries to identify a whole trajectory or a driven route from start to end, and i.a. does not only select specific locations.
2. Also, the approach has to use zero-permission sensors from smartphones and does not rely solely or mainly on GNSS-based sensors. Also, WiFi or Bluetooth is not used. The attack may use additional data such as traffic data or weather data.
3. The data was produced while driving a vehicle, and the approach takes the given environment into account. If an approach scales to

¹For an explanation of the notation, see Appendix B

also work for other means of transport, it is included as long as the first requirement holds.

4. Works were excluded that required additional tracking equipment besides a smartphone or mobile device.

Only works in English that are peer-reviewed were considered. It has to be available to the authors for further processing. No quality assessment was performed, and no works were excluded. The survey was carried out until July 2020; therefore, the literature is included until this date.

12.1.3 Relevant Findings

Table 12.1 Overview of the 15 publications identified in the SLR. The works are assigned to different disciplines.

Publication	Year	Publisher	Field
Pesé et al. [294]	2020	De Gruyter	Security and Privacy
Dimri et al. [109]	2020	ACM	Network
Waltereit et al. [389]	2019	ACM	Security and Privacy
Plangi et al. [300]	2018	IEEE	Mobile Computing
Li et al. [231]	2018	IEEE	Vehicular
Zhou et al. [427]	2017	IEEE	Systems Design
Won et al. [403]	2017	IEEE	Mobile Computing
Verma et al. [380]	2017	ACM	GIS
Narain et al. [276]	2016	IEEE	Security and Privacy
Ho et al. [172]	2015	ACM	Systems Design
Bhattacharya et al. [44]	2015	IEEE	Mobile Computing
Nawaz and Mascolo [279]	2014	ACM	Systems Design
Nawarathne et al. [278]	2014	IEEE	Systems Design
Gao et al. [137]	2014	ACM	Mobile Computing
Han et al. [163]	2012	IEEE	Network

Fifteen publications were selected given the search term and criteria from an initial corpus of 1,067 findings. We excluded two publications in favor of their

extended version. Table 12.1 lists all the selected items, including the publisher and the related field of the conference or the journal. Either ACM or IEEE published all identified work with one exception (De Gruyter); we did not select any works from SpringerLink or ScienceDirect for the final corpus because some exclusion criteria matched. The field (or category) is aggregated, e.g. “network” also includes subdisciplines such as “sensor networks”, or “mobile computing” includes sensors-related publication formats such as the *IEEE Sensors journal*. What is striking is the low representation of formats that directly address privacy and security, considering route tracking an attack. Other works present the ability to use IMU sensor to infer a trajectory or use the sensor array to optimize specific parameters such as energy consumption. From a time-range perspective, we can state that the research fields seem relatively new, with the first approaches from 2012 with a steady but low number of publications per year.

Analysis of Trajectory Inference Methods

12.2

This section will answer the introduced research questions on the basis of insights gathered from the document corpus. We present identified preconditions and sensors used in the different approaches that are briefly illustrated in the following.

Preconditions and Sensors (RQ1)

12.2.1

Relationship of approaches for trajectory reconstruction and the specific sensors used as a data source. The accelerometer is the most commonly used sensor followed by the magnetometer.

Table 12.2

Publication	Accelerometer	Gyroscope	Magnetometer	Barometer	GNSS	Speed
Bhattacharya et al. [44]	●		●		●	
Dimri et al. [109]				●	●	
Gao et al. [137]						●
Han et al. [163]	●					

continued on next page

Ho et al. [172]					•
Li et al. [231]					•
Narain et al. [276]	•	•	•		
Nawarathne et al. [278]	•		•		•
Nawaz and Mascolo [279]	•	•			
Pesé et al. [294]					
Plangi et al. [300]	•	•			•
Verma et al. [380]	•		•		•
Waltereit et al. [389]	•	•			
Won et al. [403]				•	•
Zhou et al. [427]					•

Sensors We identified that most of the works are based exclusively on data from smartphones. At some point, the works state that the data can also be gathered using other devices, such as the vehicle's CAN-bus (e.g. [389, 427]). Two works [44, 137] do not state the origin of the data, although, according to the sensors and data used, it may be feasible for smartphones to acquire the data (see Table 12.2). Another source for data is Original Equipment Manufacturer (OEM)-provided, open data-sharing platforms provided by OEMs where an OEM collects data from vehicles to provide added-value susceptible for malicious applications [294]. In terms of sensors used in the different approaches, the accelerometer is the most common, followed by the gyroscope and magnetometer. To be precise, the GNSS (including GPS) is found with high frequency. Works that employ that kind of sensor focus on optimizing, for example, the accuracy of a trajectory by interpolating the position or distance between successive GPS measurements (recall that the frequency is limited to 1 Hz in Android). Won et al. [403] uses GPS for specific situations to improve the feasibility of the approach, in contrast to previous works [404] which were totally neglected.

Preconditions and requirements Most algorithms that have been studied require certain preconditions for their execution. To place a route based on sensor data recorded within a traffic network, most approaches require map material that contains the ground truth. Alternatively, some approaches do not attempt to detect an unknown route but assume a closed-set scenario in which all routes to be taken have already been learned according to various properties. Verma et al. [380], for example, requires the knowledge of so-called point-of-concerns such as speed bumps, which is collected via crowdsourcing using a mobile companion (app). Map data includes spatial information on links between geographical locations in the two-dimensional space (i.e. longitude and latitude). Elevation data adds

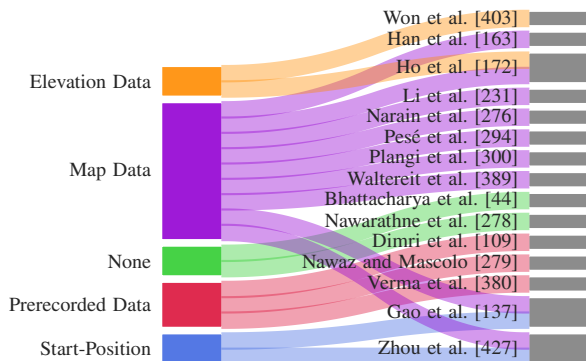


Figure 12.1

Requirements and preconditions needed by the analyzed approaches. The most common requirement is the availability of map data followed prerecorded data, although, there is no overlap between both groups.

information about the altitude profile of the route sections, i.e. the altitude. Such data is used in some works, particularly those relying on the barometer since such sensor readings can be mapped to altitude. An overview is presented in Figure 12.1. Finding a route in large road systems is challenging. Therefore, especially early works are based on knowledge of the starting point (e.g. [137]). This is a legitimate assumption in the context of PHYD since an insurer can assume, for example, external knowledge such as a driver's place of residence. As all candidates that do not start at the known starting point can be discarded, the set of potential route candidates can be significantly reduced, especially in complex road networks.

Almost all algorithms are limited in area size for route inference. In general, larger search spaces induce more complex street networks, eventually yielding many candidates for a sensor trace. Thus, it is assumed that an adversary has rough knowledge about the area in which the sensor data has been recorded. A restriction of the search space can, for instance, be made based on the IP address from which the sensor data originates. The size to which the search space must be restricted for efficient reconstruction of the route depends on the approach and varies greatly but is considered, along with accuracy and runtime, to be a significant evaluation criterion for the goodness of the approach. For example, Li et al. [231] only achieve useful results for 22 km², while Waltereit et al. [389] find routes for search areas 1200 km². Our approach significantly minimizes this limitation and allows efficient route discovery in areas of up to 4500 km².

Area size

12.2.2 Summary of Approaches (RQ2)

Trajectory inference per se is not always considered an attack, as one can conclude from the research fields presented previously. First, we look at the respective goals that the approaches pursue. Then, a standard structure is extracted, and, in the following, we look again in detail at which features support route estimation.

Goals

The works in the document corpus can be assigned to three classes depending on the pursuit of the goal. First, works may try to optimize trajectory collection by reducing the energy footprint or increasing accuracy. The other two classes both try to infer complete trajectories by mapping the sensor pattern mostly from the IMU on a map. However, one class has specific constraints that have to be met, such as knowing the start position of a route. The last class does not have such restrictions and contains an approach to mapping an unknown route on a street network.

Optimize
trajectory
quality

Some work attempts to address the problem of excessive energy consumption of the GPS sensor by no longer recording continuous traces but only activating the energy-hungry sensor for short periods of a few seconds in special situations [44, 278], e.g. in excessive heading changes. However, more extended periods of abstaining the GPS increase the uncertainty of the current location, and hence such approaches may be used in combination with dead reckoning [278]. Consequently, such approaches try to find a trade-off between energy efficiency and location accuracy. In addition, the limited frequency of the GPS and ultimately uncertainties regarding intermediate positions between location queries are addressed [278, 300]. Plangi et al. [300] addresses situations where the GPS signal fails, but also the state of the GPS drift that may occur, for example, in high-building environments. The authors fuse accelerometer and gyroscope readings to determine the probable position between measurements using map matching.

Inference
(part) routes
with
constraints

In contrast to the first class, the reconstruction of routes is the subject of this class, although some conditions are present. First, some works require knowledge of the origin of a route, i.e. a starting position to then derive potential route candidates [137, 427]. Based on this information, speed data that may originate from the accelerometer is used to generate a path consisting of turns and distances. For example, Zhou et al. [427] builds a speed model that incorporates the idea that driven speed depends on various factors such as real-time traffic, speed limits, or curve speed. Route candidates are therefore represented as different

speed models originating at the known starting point compared to the sensor data. Second, a closed-set environment may be assumed where all possible routes are prerecorded, including descriptive features [279, 380]. Nawaz and Mascolo [279] uses prerecorded angular velocities of routes as a feature that is compared with sensor data, namely gyr_z , using a proposed DTW extension called open-ended warping. Additionally, Verma et al. [380] focuses on the derivation of full-length bus lines using crowdsourcing by analyzing the sensor data stream for specific patterns similar to the ones proposed in Chapter 8 that have been recorded along with their geographical location in advance. Another stream in research tries to infer routes using the barometer [109, 403]. A user may collect pressure data in advance to generate a personal database [403] or it is obtained from publicly available resources [109]. Together, approaches are limited to prelearned data sets.

Finally, the reconstruction of a complete route from sensor data without any constraints is also present in the literature. We found approaches that use the full route to derive the starting position [163] or ultimately place a trajectory on a map w.r.t. the explained limitations [172, 231, 276, 389]. All types of sensor data can be found in the attacks found, with IMU data used in four out of five cases. Analogously to the attacks described above, based on the barometer, the measured pressure values can be converted into elevation models with the help of external databases or weather stations. Thus, a topological elevation profile can be created, which is mapped onto reference map data, as long as the errors in the measurements and models are correspondingly more minor than the entropy of the elevation models themselves. Using accelerometer data, Han et al. [163] pinpoint a smartphone within a radius of 200 m. The approach is based on the construction of a rough trajectory using the available acceleration data for map matching. Estimated trajectories are created via a probabilistic process akin to dead-reckoning that translates acceleration measurements to vehicle displacements. A device-specific, lengthy training process is required to develop the employed probabilistic model, eventually being a drawback due to limited scalability and generalization. Ultimately, some approaches do not need extended training phases as well. Such works use specific events present in the sensor stream (similar to Section 5.4) combined with domain knowledge to narrow the set of candidates within a reference map [231, 276, 389]. Li et al. [231] uses turn and respective angles detected by the magnetometer to query for matching turns in a prepared database constructed from OSM. The sequences of turns can be expanded to full routes based on a connection in the reference data, with unlikely candidates eliminated based on additional knowledge (e.g. no stop times on highways). In contrast, Narain et al. [276] derives a sequence of turns, similar

Inference full
route

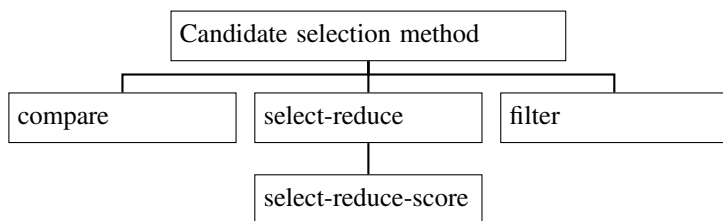


Figure 12.2 Overview of different approaches found in literature to select the final set of route candidates.

to [231], and straights, but uses a combination of accelerometer, gyroscope, and magnetometer within their approach. Furthermore, they incorporate trajectory properties such as turn angles, road curvatures, vehicle heading, and travel times to rank route candidates according to their similarity to the extracted sensor path. [389] extends both previous ideas [231, 276] by relying on turns and distances between successive turns. Distances are then used to eliminate unmeaningful turn pairs based on connections in the reference map data.

Structure

We found four independent structures underlying the approaches presented. Sensor data record an approximation of reality that deviates from measurement inaccuracies. Therefore, the identified works try to find a corresponding counterpart in their model w.r.t. such deviations. Different approaches can be used to select a route from a model that corresponds to the one recorded in reality. They are shown in Figure 12.2 and are explained in the following.

compare The first class is based on the idea of comparing the trajectory of the sensor data and potential candidates using certain quantity metrics, with DTW being used almost exclusively (for an explanation on DTW, see Sidebar F). The one where a distance or error is minimal is considered the best fitting route candidate. We found a correlation between this class and the sensors. In particular, works that use continuous data from the barometer or are based on interpreting the speed are of this class [172, 279, 403, 427]

select-reduce Sensor data is converted to objects and properties that are searchable within a database [231]. Thus, specific geographical elements are selected that represent specific cornerstones that are likely part of the route taken. The selection process may generate a large set

of elements whose size will be reduced in a successive steps using an approach-dependent predicate to fulfill requirements of efficiency and meaningfulness. For example, Li et al. [231] selects turns from a map using magnetometer-derived turn angles, connects them based on links found on that map, and then reduces the improbable stop times of the crafted routes in road segments. In this case, a list of equally probable candidates for the recorded sensor trajectory is provided.

select-reduce-score This class is an extension of the previous class that considers that sensor patterns may fit several elements in map data, although an adversary may be interested in the best fit. Hence, route candidates have to be ranked or scored to yield the most probable trajectory after finishing the select-reduce step. Multiple elements have been proposed, starting from the turn angle similar to the road curvature [276] and distance [389] and others [172]. ; for example, the distance between successive elements can be constrained and must be met [389]

filter By employing standard filtering techniques such as Kalman, low pass, or moving average, this class is mainly found in the first category of approaches, that is, the optimization of trajectory extraction using GPS [300].

Features

Specific elements are extracted from sensor data to find a route in a street network by searching for these common features. Different works propose multiple features attached to or represent a specific element in a map graph. Now we briefly list them.

- ▶ **Turns** are specific structures in a road network that allow a direction change of a specific angle. Some works only rely on the evoked heading change and use a sequence of direction changes to find on a map [389]. Others [231, 276, 279] use the entropy of a turn (i.e. its turn angle of $[0^\circ, 360^\circ]$) to narrow down potential turn candidates. [300] only use turns to correct heading; hence, do not perform any selection based on this. Waltereit et al. [389] also considers the false positive or false negative detection of turn in their approach.
- ▶ A road's **heading and curvature** can be extracted from the map data using geographically annotated nodes as we have shown in Section 5.5. These properties are compared to the gyroscope readings [276]. The

heading is used to reduce the number of turn candidates, while curvature is used during scoring to select the most feasible route candidate.

- ▶ Next, **distances** are used in combination with fixpoints such as turns to reduce or score route candidates [278, 389]. This feature is difficult to acquire for the reasons illustrated in Section 2.3 and Chapter 8.
- ▶ While the curvature is a trajectory in the flat plane, **elevation** considers the altitude profile of a taken path and the map data, respectively [172]. Depending on the resource, the rasterization, i.e. artificial areas having the same height, is relatively sparse.
- ▶ In addition, the **travel time** is similar to distance but more robust to derive and is used to remove specific road segments that are not suitable as part of a candidate route [276]; Zhou et al. [427] uses it to calculate the average speed based on current traffic conditions.
- ▶ Last, additional **descriptive features** are found in various approaches. Examples are road works, speed bumps [380], and traffic lights [276] as well as speed limits [231]. Such features are similar to those proposed in Sections 5.4, 5.5 and 8.4.

It should be mentioned that depending on the property used, a certain quality of the sensor data is required, and its deflections have to be handled accordingly. For example, distance can be accurately determined using GPS, but accuracy suffers as soon as a derivation is made based on the accelerometer.

12.3 Evolution of Trajectory Inference

In the literature, various approaches have been presented that are suitable for inferring the route driven using sensor data. As we have learned, methods are distinguishable based on if they require knowledge about the starting point of a route or approaches or if they only need to know the approximate region as a precondition. Such attacks also depend on the region's size in which a route is searched since more extended regions usually indicate more complex road networks. The sum of theoretical routes to take within the road network increases accordingly. Two aspects are related to this. The more routes that can be found as possible candidates, the harder it becomes to determine the correct route within the set of candidates. Furthermore, the runtime is also of interest, with shorter ones desirable to maintain feasible detection performance.

We now present our approach to trajectory inference that combines the ability to search for a route in a wide-ranging geographic area with fast runtimes. Our approach uses sensor data gathered by smartphones and is limited to data from zero-permission sensors to stress the threat to user privacy. Hence, we consider it an attack that can be embedded in the context of UBI. The inference attack is based on the idea that a trajectory is a sequence of turning directions and distances similar to Waltireit et al. [389]. Consequently, Funke and Storandt [134] state that the problem of map matching can be reduced to retrieving the so-called *path shapes* in a street network. The path shape is defined as a relative view of the trajectory of a vehicle of the following form $(d_1, \alpha_1, d_2, \dots, d_{n-1}, \alpha_n, d_n)$ where α_i is a turn with a specific turning angle. d_{i-1} and d_i are the distances driven before and after the turn. Such a trajectory may produce a unique pattern that can be found in a street network like OSM.

Our
approach

The uniqueness of a pattern depends on several aspects. Although the number n of elements in a path shape may be relevant, individual elements are also essential. The type of curve can be used as a curve is defined by its angle of turn, denoted as α . Since vehicles follow a road network, the measured turn angle is dictated by the road network and, therefore, may be found in the map data. Its entropy may vary depending on the road network, though [48], hence we also include as many additional features (see Section 12.2.2) as possible. The orientation of the driving path can limit the search space, which can also reduce the number of potential routes [231]. Furthermore, distances between curves can be used to constrain sequences of curves [389]. A detached consideration of distances is not useful. However, the sections traveled allow insight into the curvature of a route, indicating if a road tilts more to the left or right, as proposed by Narain et al. [276]. This information may yield a high entropy, which can be used to reduce the search space further. Finally, the route search can incorporate additional knowledge found in sensor data. This includes stopping times at traffic lights or traffic circles (c.f. Chapter 8) that are also present in OSM.

Uniqueness
of a
trajectory

Information on distances and turns can be derived from the accelerometer *acc* and the gyroscope *gyr* as presented in Chapter 4. Furthermore, the turn angle α and the curvature can be determined using consecutive readings of gyr_z .

Data source

Threat Model

12.3.1

Two parties are present in the given scenario: The driver and the adversary. The adversary's objective is *to determine the driver's trajectory without relying on standard location services* such as GPS, WiFi, or mobile cells, since access to these data is usually restricted (c.f. Section 3.1) but uses sensor data from

Adversary

zero-permission sensors. Therefore, he relies on data from the accelerometer, gyroscope, and magnetometer that are present in most modern smartphones. In the case of UBI he may use the transmitted sensor data from clients, although this side-channel attack is not limited to this context. The needed data can be gathered using any honest-but-curious app that may be available from the app store in a disguised shape, as it is common for such attempts (see Chapter 10). Data collection can be triggered by movement based on the accelerometer [231, 276] to reduce the impact on performance or narrow the data stream. It is sufficient for an adversary to gain access to the sensor data to process it *a posteriori*. Hence, we assume that the smartphone is present during trips to collect the respective data and then submit them to a colluding server.

External area
knowledge

Another assumption is that the attacker needs to know the approximate area, such as the city or region, of the victim in order to make the route search appropriately efficient. For example, it is possible to make an approximate estimate of the region by performing a geo-lookup of the device's IP address during data submission. Since our attack is intended to be efficient for large areas, such an estimate should be sufficient. No further information *a priori* about the victim is necessary. In particular, any route within the geographical region and the respective street network is equally likely to be taken. For example, the attacker does not incorporate heuristics about road usage into his guess.

Fixed
smartphone
position

Last, since the attack is based on sensor data from a smartphone, it is required that the device is in a fixed position (e.g. pocket, cup holder, charging tray) while recording data to only measure changes in sensor readings due to driving. This requirement is similar to any use case that processes sensor data from smartphones in a mobile environment (e.g. Chapter 8). However, this is a meaningful and sustainable assumption since laws in most countries prohibit using smartphones while driving (even when stopped e.g. at traffic lights) for safety reasons.

12.3.2 Challenges

Similar to the previous attack on driver identification (see Chapter 11), trajectory inference attacks are simultaneously subject to several challenges to be applicable in a real-world scenario.

Sensor
accuracy

We discussed the varying accuracy of sensor data from device to device and manufacturer in Section 2.3. In particular, noise and errors within the sensor data stream have to be dealt with to accurately depict a path shape of a trajectory. External factors such as the environment may also implicate the sensor accuracy.

For example, the magnetometer is susceptible to magnetic distortion induced in a vehicle e.g. through the speaker [187]. Additionally, the road quality or the vehicle itself may produce additional noise caught by the accelerometer or gyroscope.

As depicted in this part of the work, different drivers have different driving styles. Individual behavior and traffic conditions inflict the sensor data gathered within the attack, eventually resulting in different sensor readings even for the same road segment. As the approach is based on path shapes, turns are of particular interest. However, detecting these is difficult, especially on wide and curvy roads. The distinction between a slight turn or a curvy road is closely related to driving behavior but is vital in route recognition. We have seen similar challenges within the detection of traffic circles in Chapter 8. The detection problem is further complicated by additional noise that is introduced into the FPD by changing lanes or overtaking other vehicles, braking, and accelerating [276].

Driver
behavior

Recent map data with an accurate street network to successfully project path shapes on this reference material is crucial for detection. This includes the completeness of the map data to extract the required information for the attack, such as turn angles. Since our data source, OSM, is based on crowdsourcing, the quality, informativeness, and up-to-dateness strongly depend on volunteers. Although there are specific guidelines on how particular network elements must be modeled, the quality of map data for a specific region differs. Especially wide or multi-lane roads have to be treated separately because their modeling often differs from reality, as we have shown in Section 5.5. Two challenges are amplified here, but, as already mentioned, especially such roads allow for a broad bandwidth of driving behaviors and, ultimately, sensor patterns. In OSM, for example, multi-lane roads are represented as single lanes unless they have structural separation. We also stress this in Section 5.5. As a result of this deviation, sensor data cannot be mapped to the road network, making route recognition difficult.

Map data
quality

Building Blocks

12.4

In this section, we first introduce the sensor data that is the fundamentals of the attack. It is comparable to the data used in the context of UBI, which means that the attack can be applied to this area. Next, the attack is formally prepared by outlining an appropriate model that efficiently and effectively supports route search. Then, the process is outlined of how the data from a data source (OSM) is appropriately prepared.

12.4.1 Sensor Data

The foundation of the attack is provided by the unprotected sensor data of IMU, which generates a series of multivariate time series data \mathcal{M} . We adopt the notation of measurements $m_l \in \mathcal{M}$ here, which was introduced in Section 5.2.

$$m_l = \langle t, \vec{a}cc, \vec{g}yr, v, loc, h \rangle$$

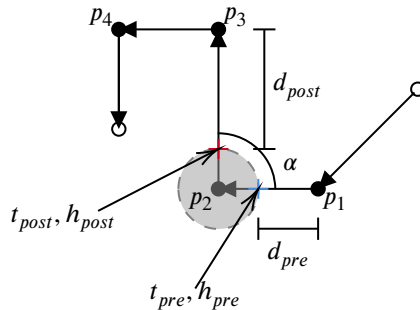
The attack is based on path shapes that require turns and distances to be extractable from a measurement. Turns may be detected using the gyroscope as explained in Section 5.4. Recall that due to the limitations and error-proneness of the magnetometer, a heading h cannot be used consistently for the detection of curves. Due to this inaccuracy, it is only used for the approximate determination of vehicle orientation but not for the detection of curves. In addition, we calculate the distance d_i covered by between two measurements m_i and m_{i-1} based on v using $d_i = (v_i + v_{i-1})/2 \cdot (t_i - t_{i-1})$ and add that to each measurement m_l . Consequently, v may be estimated based on methods initially mentioned in Chapter 4. Experiments have also shown that the distance derived using solely IMU data is from varying accuracy. This fact will be incorporated into the attack adequately. Each measurement is subject to data cleansing, such as removing outliers (e.g. peaks in velocity that result in unrealistic distances between measurements). Since the attack is a zero-permission sensor attack, access to GPS data is not required. In a realistic scenario, we assume \emptyset for any loc , although we collect the location for measurements in order to capture the ground truth for evaluation purposes. We also apply smartphone-to-vehicle alignment techniques that are described in Chapter 4 to allow arbitrary positions within the vehicle.

Recapitulation of elements from road

The attack will also take into account special patterns in the sensor data that have high information content because of their rarity. These can significantly reduce the search space depending on the environment and increase the detection quality. In particular, elements identified in Chapter 8 are used. The elements in question are traffic circles and traffic lights. Both of them are also included in the map material of OSM and therefore fulfill the important requirement of data availability. The algorithms presented in the corresponding chapter are applied. Traffic circles are detected by the unique M/W pattern, whereas the speed profile and idle phase support the detection of traffic lights.

12.4.2 Model

In the following, we present the model that allows the effective search of candidates for the driven route.



Example path \mathcal{P} with $n = 4$ path events p_i .

Figure 12.3

Constructing Paths

We now formally define a *path* based on the idea of path shapes [134] but extend it to also take into account the covered distance. The path \mathcal{P} represents the relative movement of a vehicle based on recorded sensor data measurements \mathcal{M} as a sequence of turns (or traffic circles) with distances, the so-called *path events* p . Let P be a function that generates them based on measurements: $P : \mathcal{M} \rightarrow \mathcal{P}$, then a \mathcal{P} is an ordered sequence of n path events (p_1, \dots, p_n) . Each path event p_i is a vector defined as

$$p_i = \langle \alpha, t_{pre}, t_{post}, d_{pre}, d_{post}, h_{pre}, h_{post} \rangle$$

with α being the angle of the turn starting at t_{pre} with incoming heading h_{pre} and ending at t_{post} with outgoing heading h_{post} . d_{pre} and d_{post} denote the distance to the previous and following turn, respectively. Figure 12.3 illustrates the idea of a path \mathcal{P} with $n = 4$ path events p_i .

Street Network

Similar to previous works [276, 389] the search for \mathcal{P} is executed within a directed graph that represents our street network based on $G = (V, E)$ (c.f. Section 5.5). We construct a street network $\mathcal{G} = (S, C, A, B, L, C)$ that holds the required information to efficiently lookup a given path \mathcal{P} . S is the set of unidirectional road segments s_1, \dots, s_m with a well-defined, single start and endpoint. A road segment is deterministically traversed from start to end. C is the set of all connections c_1, \dots, c_n . A road connection c_i enables the transition between two road segments (s_u, s_v) by defining a connection, i.e. $s \sim v \in V$. The connection is likely due to a shared geographical point. It is represented as a tuple

$c_i = (s_u, s_v)$ with $s_u, s_v \in S$. A road connection can be considered an intersection, i.e. $c \sim e \in E$.

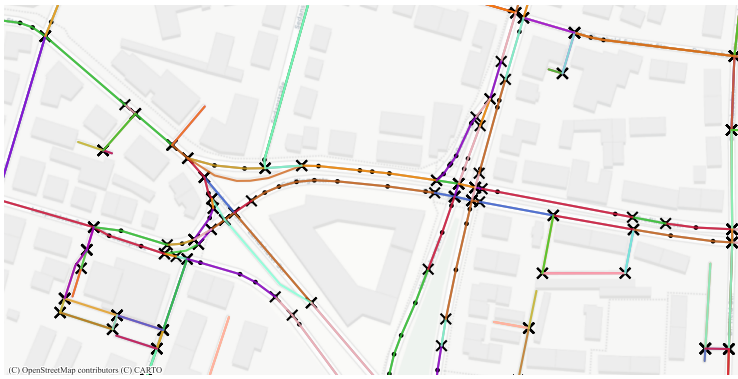
A road segment is a vector in the 2-dimensional space and thus a specific direction. For example, a path runs from north to south. This orientation can change at connections either left, right, straight, or reverse (in terms of a u-turn) but not within a segment. Let A be a function to calculate the angle between two segments meeting at a connection c_i . That angle allows to refine the type of the connection further and not only binned orientation changes. We consider a connection a *turn*, if a threshold is exceeded, i.e. $|A(c_i)| > 30^\circ$ holds. Thus, set the of all turns is defined as $\hat{C} = \{c \mid |A(c)| > 30^\circ, c \in C\}$. Consequently, that implicates everything not being a turn being a *straight* connection, resulting in the disjoint set $\tilde{C} = C \setminus \hat{C}$.

We are now loosening the strict separation of turns and straights by introducing so-called *slight turns*. These take into account the inaccuracies in driving behavior already mentioned. First, we define a straight connection c_j as a connections where $|A(c_j)| \geq 30^\circ \wedge |A(c_j)| \leq \theta_S$ holds. θ_S is called slight-turn threshold. This marks a transitory zone from straight to turn and hence is constrained to be $\theta_S \geq 30^\circ$ (e.g. $\theta_S = 60^\circ$). A slight turns yields a turn as well as a straight connection, i.e. $\tilde{\tilde{C}} \leftarrow c_i, \hat{C} \leftarrow c_i$. Hence, $\tilde{\tilde{C}}$ and \hat{C} are not disjoint anymore. This enables handling overlapping situations explained in Section 12.3.2. In particular, handling specific connections in hybrid way takes into account different driving styles and prevents missing undetected turns eventually degrading trajectory inference. A 's output is aligned with the compass direction with negative values indicating a movement to the left.

B is crucial for the attack as it is used to reduce the search space. It defines the orientation of a road segment in relation to the north after leaving the road connection that connects it to other road segments (i.e. bearing). We will emphasize the benefits when explaining the attack in Section 12.5. Two additional utility functions, L and C , are used to calculate the length of segment s_i and its curvature, respectively.

12.4.3 Network Construction

We formally defined the street network \mathcal{G} in the previous section that is to be constructed based on OSM data (c.f. Section 5.5). Recall that the OSM street network is based on nodes and ways, i.e. $G = (V, E)$. However, a way is a semantically and syntactically connected sequence of nodes that are used to describe the shape of that specific way. A way is composed of *way segments* that represents the shortest straight connection between two nodes. They are used to



Section of the processed road network. Shown are the extracted road segments \mathcal{Q} and road connections \mathcal{Q} based on data from OSM.

Figure 12.4

describe the shape of a way. This segment is an element of \mathcal{V} . The direction of a way is also of interest since G is a directed graph. Consequently, we create two way segments for any way other than single-direction ways (e.g. one-way streets). Metadata such as the street type and the speed limit are extracted from OSM and stored along. Intersections within the OSM street network are identified by looking for edges E that connect more than two vertices \mathcal{V} . They are relevant for deriving the set \mathcal{S} .

Detection of Links between Intersections

First, it is essential to know that only intersections allow one to take different routes within a street network. In particular, a way in OSM can skip an intersection. Therefore, a way is not semantically considered a link between intersections.

Differences from OSM

Moreover, we consider every $e \in E$, i.e. way segment, as a potential connection between two intersections. For efficiency reasons, however, the way segments are to be combined which do not represent information about the structure of the graph other than connections between two nodes. This procedure will allow a fast search for the link between two arbitrary intersections. The composition of way segments is the basis for the road segments, which will be created after preparation in the next phase. The difference becomes clear with the help of Figure 12.4; each colored line represents a road segment. One can see that these often span multiple nodes that are represented as circles but are still shape-aware by respecting way segments.

Projecting way segments to road segments

Implementa-
tion details

On this basis, we generate a list of all connections between two intersection pairs. This is achieved by starting at an intersection and aggregating successive way segments until another intersection is reached. The traversal is performed by using the endpoint of a way segment as the start of the next segment. Eventually, a link starts at an intersection and ends at one. Dead-ends are taken into account by handling them accordingly since a second intersection will not exist. To calculate a road segment's curvature later, we still store each way segment, in particular nodes, between two intersections. The process is completed when each intersection identified previously has been processed. Traffic circles are excluded at this point.

Extraction of Road Segments

Definition

A road segment is defined as the direction-aware link between two connections or intersections. They were identified in the previous step. A road segment is composed of way segments (i.e. legs) that are merged to allow efficient traversal within our attack. It may bend excessively between both ends, as shown in Figure 12.4 where the colors indicate a whole road segment.

Finding road
segments

Let a function \tilde{A} exist to calculate the angle between three points² and \tilde{L} to calculate the geodesic distance between two points [201]. A road segment is an ordered sequence of n OSM nodes with geographical information: $s_i = (v_{i,1}, \dots, v_{i,n})$ based on the way segments on which it is constructed by. Then there should be no combination of the nodes $\blacksquare = (v_m, v_{m+1}, v_{m+2})$ with $\tilde{L}(v_j, v_{j+1}) > 15 \text{ m} \wedge v_j \in s_i$ where $\tilde{A}(\blacksquare) > 30\%$. Otherwise, that segment is considered to contain a turn, which is likely to happen in real-world data sets.

Refinement
of road
segments

The circumstance that a road segment contains a turn violates the model. Therefore, it must be eliminated by splitting that road segment at the point where \tilde{A} has its maximum. We consider additional requirements during split that the turn is ensured to not split within a turn composed of multiple nodes in OSM. A split replaces the existing s_i with two new road segments. This process is repeated until each road segment satisfies the no-turn constraint. This can be seen in Figure 12.4 where a road segment is not necessarily limited by two intersections, i.e. e that connect more than two vs .

Properties of
road
segments

Our model \mathcal{G} allows to calculate the length and curvature of any given $s \in \mathcal{S}$ using the two functions L and C . L calculates the length of a road segment by aggregating the lengths of all the way segments. C is determining the curvature

² For two-dimensional points A, B, C , $\tilde{A} = \arccos\left(\frac{AB^2 + BC^2 - AC^2}{2 \times AB \times BC}\right)$ may be such a function



a) Ongoing turn. The ongoing turn (colored) spans multiple nodes resulting in a miscalculation of the turn angle. The cross marks the actual turn according to definition.

b) Multileg turn. A turn may proceed over multiple segments in reality. Each color represents a segment that is separated by a connection (i.e. turn) marked with a cross.

Different challenges present in \mathcal{G} . The challenges have to be considered when modeling a street network \mathcal{G} to match it against driven paths \mathcal{P} .

Figure 12.5

by aggregating the angles for each adjacent node triplet. In addition, **B** allows to calculate the orientation or bearing when entering a road segment w.r.t. the model.

Connection of Road Segments

Once all the road segments \mathcal{S} have been identified, they must be connected accordingly so that \mathcal{G} contains all possible paths that a vehicle can take. Thus, we now form the set \mathcal{C} that not only holds all intersections (as defined by OSM) but also turns.

The angle of a connection $c_i = (s_u, s_v)$ can be determined by selecting one point in the incoming segment s_u and one in the outgoing segment s_v as well as the connection itself. Subsequently, the angle can be calculated by **A** for these three points. Based on the angle, the direction of the turning maneuver of the intersection can also be found as described (left, right, straight). However, this simplistic approach is widely used in the literature [231, 276, 389], but is limited in its feasibility in matching real driving patterns in a street network model. As the set of connections \mathcal{C} is essential to match a \mathcal{P} , its completeness and correctness are crucially important. Now we explain the main derivations in detail.

Orientation
changes
during
traversal

Ongoing
turns

OSM uses way segments containing multiple nodes to describe the shape of the underlying street. The simplistic approach ignores the fact that, commonly, more than three nodes may comprise a turn. This case is depicted in Figure 12.5a where only the highlighted points are considered to be the turn. A calculation based on the two closest nodes will not yield the correct angle. To also consider the challenge of differentiating slight turns and curvy roads, we extend the search space by including composite road connections that may represent undetected turns depending on the driving behavior. Therefore, we greedily aggregate neighboring nodes on the start and target segment, respectively, until the distance between two neighboring nodes is above 10 m, the slope changes, or the angle surpasses 10° .

Composite
turns

In a complex road network, intersections can occur in greater numbers and in confined spaces, especially in urban areas. The model created so far considers connections in isolation. Assume that the segment connections $c_1 = (s_u, s_v)$ and $c_2 = (s_v, s_w)$ are movements to the right with $A(c_1) \geq 15^\circ$ and $A(c_2) \geq 15^\circ$, respectively. Furthermore, the segment s_v is a short segment with $L(s_v) < 30$ m. Depending on the steering maneuver, the two immediate connections c_1 and c_2 may be identified as a single (slight) turn. Figure 12.5b illustrates this issue. Each color and shape is a valid road segment according to the definition. However, the middle segment may not be separated by dedicated turns in a realistic scenario. To account for such cases, we include an extra connection $c_3 = (s_u, s_w)$ that is the composite of c_1 and c_2 in terms of shape, length, and angle. However, the definition of turn still is applicable; thus, c_3 is only a valid turn if its angle is above the threshold of 30° . This yields a larger C but allows for a more thorough evaluation of possibly detected bends on curvy roads.

Unique
shaped turns

In addition to previous work, we also include traffic circles in the attack as they have a unique pattern (explained in Chapter 8), significantly reducing the search space once such a pattern is present in \mathcal{P} . Traffic circles are identified based on a tag in OSM. A traffic circle is handled as a standard intersection, and a connection is also marked as a traffic circle in our street network \mathcal{G} . Having both information enables the search in C even if the sensor pattern (for a traffic circle) is not recognized correctly. We explain in detail the reasons for missing that pattern in Section 8.6.

12.5 Inferring Trajectories From Sensor Data

Starting
point

We have constructed the street network $\mathcal{G} = (S, C, A, B, L, C)$ that provides information on i.a. connections. At each passed connection $c_i \in C$, the route of

a vehicle may change due to a steering event. We collect steering events in the form of turn events $p_i \in \mathcal{P}$ that are obtained from \mathcal{M} .

To match a given trajectory \mathcal{P} in \mathcal{G} , we derive potential route candidates \mathcal{R}^* . A route candidate $r_i^* \in \mathcal{R}^*$ is considered a sequence of n turns $r_i^* = [\hat{c}_1, \dots, \hat{c}_n]$ present in \mathcal{C} . Consequently, for each turn event p_i , we identify similar turns within the set of all turns $\hat{\mathcal{C}}$. This yields a set of turn candidates $\hat{\mathcal{C}}_i^* \subset \hat{\mathcal{C}}$. Then all consecutive turn candidate sets $\hat{\mathcal{C}}_i^*$ and $\hat{\mathcal{C}}_{i+1}^*$ are intelligently connected w.r.t. the given \mathcal{G} to form route candidates \mathcal{R}^* . Ultimately, a $r_j^* \in \mathcal{R}^*$ must be chosen that is considered the best match for a given path \mathcal{P} .

Overview of the approach

We thoroughly explain each step in the following section. First, we explain how to retrieve turn candidates $\hat{\mathcal{C}}^*$. Then the candidates are exploded to full route candidates \mathcal{R}^* . Lastly, multiple scores are incorporated to rank $r_j^* \in \mathcal{R}^*$ to derive a decision on the best candidate for the matching route for \mathcal{P} .

Structure

Retrieving Turn Candidates

12.5.1

The set of turn candidates $\hat{\mathcal{C}}_i^*$ for each $p_i \in \mathcal{P}$ is constructed by selecting turns from $\hat{\mathcal{C}}$ that have a specific similarity to the path event. The set of turn candidates should contain as few elements as possible to allow efficient processing and manageable size of \mathcal{R}^* but must contain the ground truth, i.e. the actual turn. Therefore, an efficient filtering approach is needed.

Extensive candidate set size

Recall that a path event is defined as $p_i = \langle \alpha, t_{pre}, t_{post}, d_{pre}, d_{post}, h_{pre}, h_{post} \rangle$. To be a reasonable turn candidate, any connection $\hat{c}_j = (s_u, s_v)$ must satisfy three constraints given to be included in $\hat{\mathcal{C}}_i^* \leftarrow \hat{c}_j$ for p_i , that is, the magnitude of the turn angle α , the incoming and outgoing orientation w.r.t. the heading h , and the distance d to related turns.

Path events

The turn direction must be respected when selecting a turn from the set of all the turns. Furthermore, the angle of the turn $A(\hat{c}_j)$ must be similar to $p_{i,\alpha}$. However, it cannot be assumed that the measured angle corresponds to the reference angle in OSM. Therefore, the following condition is applied to meet the similarity: $|p_{i,\alpha} - A(\hat{c}_j)| \leq \theta_A$. θ_A allows to match turns even when angle by OSM is different from the sensor data. This is required due to two aspects. Either measurements of the gyroscope may have inaccuracies even though it has been proven to be reliable [276], or the angle derived from OSM and real-world trajectories are divergent. Since it is crucial to not miss the correct turn, it is recommended to choose a rather relaxed value for θ_A .

Similarity of turn angle

Equality in orientation Per definition, a turn \hat{c}_j allows a heading change w.r.t. the geographical orientation. Based on Narain et al. [276], the orientation after a turn event $p_{i,h_{post}}$ will be used to constrain the selection of meaningful turn candidates. Hence, the rough orientation after passing a turn must match the outgoing's segment direction $\mathbf{B}(s_v)$ of a turn: $|p_{i,\alpha} - \mathbf{B}(s_v)| \leq \theta_M$. Apparently, magnetometer readings have to be normalized within the compass range of $[0^\circ, 360^\circ]$ to be comparable with a segment's orientation since the IMU's output is unbound. We also use θ_M as an error threshold for the magnetometer. As explained in Section 2.3 the magnetometer is susceptible to disturbances that are incorporated into the attack.

Limitation of length \mathcal{P} contains all meaningful turns as single path events p_1, \dots, p_n . Therefore, the distance between each turn p_i and p_{i+1} cannot contain any additional turn. Road segments are designed with this constraint in mind so that the length of a road segment (as of L) is the distance between its bounding intersections. Hence, additional filtering can be applied while selecting reasonable candidates. This is why we compare the distance to the preceding and subsequent turns (intersections) similar to Waltereit et al. [389]. Let us use the following predicate that must be satisfied for each turn candidate \hat{c}_j to be included in \hat{C}_i^* as a valid candidate for p_i : $L(s_u) \leq (1 + \theta_S) \cdot p_{i,d_{pre}} + \tau_s \wedge L(s_v) \leq (1 + \theta_S) \cdot p_{i,d_{post}} + \tau_s \cdot \theta_S$ represents an error threshold for the estimation of rough distances that occurs due to the nature of IMU data. This is due to the fact that we derive distances from a combination of *acc* and *gyr*, and therefore the distance between two turns might be inaccurate in some terms (see Chapter 4). The purely relative consideration exacerbates the problem of exact turn detection, in particular, the exact turn point, especially in the case of immediate turn sequences. Unlike Waltereit et al. [389], an absolute error term τ_s is also introduced to handle such cases.

Outcome This step yields a set of turn candidates \hat{C}_i^* for each element $p_i \in \mathcal{P}$. Traffic circles are also included in each respective set of candidates but eventually significantly reduce this set in size due to the given to-be-met constraints.

12.5.2 Exploding Candidate Routes

Multiple candidate sets Once the respective candidate sets $\hat{C}_1^*, \dots, \hat{C}_n^*$ for a path with n path events have been created, the existing route candidates w.r.t. \mathcal{G} are about to be constructed. It is assumed that two successive candidate sets are surjective, although, due to the isolated identification of turns in the previous step, that might not be the case yet.

Processing surjective sets The surjective property is subject to this processing step. The combination of two successive candidate sets \hat{C}_i^* and \hat{C}_{i+1}^* is assumed to be non-injective. We

are about to find the intersection $\hat{C}_{i,i+1}^* = \hat{C}_i^* \cap \hat{C}_{i+1}^*$ between both sets. The intersecting elements are defined in such a way that there is *at least* one link from each $\hat{c}_a \in \hat{C}_i^*$ to any $\hat{c}_b \in \hat{C}_{i+1}^*$. A link is considered a directed sequence of l segments between two turns denoted as $s^* = [s_1, \dots, s_l]$. If such a link exists, the pair of turns (\hat{c}_a, \hat{c}_b) intersects and is included in the set of results $\hat{C}_{i,i+1}^*$.

G and ultimately \mathcal{G} is a fully connected graph; therefore, a sequence could exist between any given combination of two turns. However, in order to include a turn combination (\hat{c}_a, \hat{c}_b) and its sequence of segments $s_{a,b}^*$ as a potential part of a route candidate r^* , the covered distance $L(s^*)$ (that is $\sum_{s \in s^*} s$) must be within the interval $\left[(1 - \theta_S) \cdot p_{i,d_{post}} - \tau_s, (1 + \theta_S) \cdot p_{i,d_{post}} + \tau_s \right]$. Furthermore, the segment sequence between two turns has to be linked by straights; hence $\forall s \in s^* \exists c \in \hat{c} : c = (s_i, s_{i+1})$ must hold. Turns as connectors are excluded because they should have been present in the sensor data. That said, the relevance of slight turns is shown.

Connection of candidates via straights

Previous work has introduced a pairwise matching approach to the problem of finding intersections between n sets that reduce the runtime to $\log_2(n)$ [231, 389]. A tree structure allows parallel processing of candidate pairs: When processing each combination of the n -ary Cartesian product $\hat{C}_1^* \times \dots \times \hat{C}_n^*$, the problem is reduced to processing two-set Cartesian products $\hat{C}_i^* \times \hat{C}_{i+1}^*, \dots, \hat{C}_{n-1}^* \times \hat{C}_n^*$ in parallel. In the next step, intersecting combinations such as $(\hat{C}_i^* \cap \hat{C}_{i+1}^*) \times (\hat{C}_{i+2}^* \cap \hat{C}_{i+3}^*)$ are analyzed w.r.t. the shown constraints.

Tree-based linking process

Finding s^* , i.e. the sequence of segments that connects two turns is an expensive operation of quadratic processing time since the traversal of \mathcal{G} is elaborate. Therefore, attempting to perform this process with as few elements of Cartesian products as possible is desirable. We apply the following optimizations:

Optimization of the explosion process

- ▶ Waltereit et al. [389] proposes that all turns $\hat{c}' \in \hat{C}$ are identified preemptively whose distance and approximately correspond to the distance $p_{i,d_{post}}$ and thus are reachable from \hat{C}_i^* . Originating from each turn candidate in \hat{C}_i^* , we start traversing the street network \mathcal{G} and collect each visited c' in a set $C_{i+1}^{*'}$ on that path. Traversal stops if the length of the aggregated road segments exceeds the approximate distance to the next turn $p_{i,d_{post}}$. The second candidate set can be reduced to turns that are in the original set and in the visited set, $\hat{C}_{i+1}^* \cap C_{i+1}^{*'}$ (the intersection of sets in this case is used in its original understanding, i.e. a set of common elements of sets).

- ▶ Knowledge about a path further reduces potential candidates that would need mapping in a later iteration. Given a sequence of road segments s^* between two turn candidates, we discard any tuple that contains duplicates, i.e. the same road segment is traversed more than once, as this can happen especially on long trajectories.
- ▶ Next, we can incorporate the recorded heading change $y_i = p_{i+1,d_{pre}} - p_{i,d_{post}}$ and the expected heading change $y'_i = \sum_{s \in s_i^*} \mathbf{B}(s)$ where s_i^* represents the segment sequence from a connection i to a connection $i + 1$. The filter rule to exclude a turn pair from $\hat{C}_{i,i+1}^*$ based on the heading prospection is now defined as $|y_i - y'_i| > \theta_H$.

Outcome The process is repeated until all turn pair candidate sets $\hat{C}_1^*, \dots, \hat{C}_n^*$ have been processed and merged accordingly in the iterative approach. The final set is defined as the set of route candidates $\mathcal{R}^* = \hat{C}_{1,\dots,n}^*$ that contains fully connected routes.

12.5.3 Ranking of Candidate Routes

Viewing and comparing route properties

The decision to select the most likely route candidate $r^*_i \in \mathcal{R}^*$ is supported by five similarity metrics. Normalized metrics, which are i.a. based on the selection criteria introduced previously, consider different aspects of the route. The more semantically similar a candidate r^*_i is to the actual route r , the lower a penalty score χ_Σ based on each metric χ and the corresponding weight λ , similar to Narain et al. [276]. The penalty score incorporates information from \mathcal{G} and compares it to \mathcal{P} obtained from \mathcal{M} . The rating is performed for each candidate in the set \mathcal{R}^* .

$$\chi_\Sigma = \lambda_A \chi_A + \lambda_D \chi_D + \lambda_H \chi_H + \lambda_C \chi_C + \lambda_{TL} \chi_{TL}$$

Turn angle similarity

According to previous work [231, 276], we calculate the similarity of the turn angle χ_A between the recorded path p_1, \dots, p_n and a route candidate $\hat{c}_1, \dots, \hat{c}_n$. Eventually, the average difference of a sequence with n turns is calculated by

$$\chi_A = \frac{1}{n} \sum_{l=1}^n | (p_{l,\alpha} - \mathbf{A}(\hat{c}_l)) |$$

Turn distance similarity

Our model assumes no turn within a road segment sequence that connects two turns. Similar to previous works [137, 138, 389, 427], we calculate the deviation

of the distance between two successive turns as we measure it in \mathcal{G} and \mathcal{P} . The distance between two turns in \mathcal{P} is given by a path section's $p_{l,d_{post}}$ property. This is compared to $L(s_{i,i+1}^*)$, i.e. the distance between turn i and $i + 1$. This results in the following equation:

$$\chi_D = \frac{1}{n-1} \sum_{l=1}^{n-1} | (p_{l,d_{post}} - L(s_{l,l+1}^*)) |$$

We constructed a score χ_H to compare the similarity of the heading change between two turns. Hence, we present the turn pair matching algorithm by using y_i and y'_i introduced in the previous section. This metric is a generalized view of the heading change without incorporating information about how the change is developing but allows for differentiation between straighter from curvier roads. For example, a highway change may be lower than rural roads. Hence, we define

Heading
change
similarity

$$\chi_H = \frac{1}{n-1} \sum_{l=1}^{n-1} | (y_l - y'_l) | = \frac{1}{n-1} \sum_{l=1}^{n-1} | (p_{l+1,d_{pre}} - p_{l,d_{post}}) - \sum_{s \in S_l^*} (\mathbf{B}(s)) |$$

The exact curvature between two turns is considered to be of high relevance within the selection process, as other work has already stated [276]. The curvature can be considered as a sequence of immediate heading changes. Therefore, this metric reflects the heading change similarity metric as more nuanced. In the previous metric, the heading change was determined by two points, one after turn \hat{c}_i and one before turn \hat{c}_{i+1} . The idea of the segment sequence curvature similarity metric is to increase the number of observation points by determining the heading after each partial segment $s_{i,j} \in s_{i,i+1}^*$ between those two turns \hat{c}_i and \hat{c}_{i+1} using \mathbf{B} and the distance traveled accordingly. Based on the distance traveled, two $(m_o), (m_p) \in \mathcal{M}$ corresponding to the measurement at the beginning as well as at the end of a road segment $s_{i,j}$ can be determined. All measurements $\mathcal{M}_{i,j} = [m_q \in \mathcal{M} \mid m_{o,t} \leq m_{q,t} \leq m_{p,t}]$ corresponding to the measured values falling in this interval. In the context of the measured values, a change of direction and finally, the heading can be determined based on $\mathcal{M}_{i,j}$, which is denoted by $\mathbf{B}(\mathcal{M}_{i,j})$ for the sake of simplicity. Finally we can construct the metric χ_C as:

Segment
sequence
curvature
similarity

$$\chi_C = \frac{1}{z} \sum_{i=1}^{n-1} \sum_{j=1}^{l_i} | \mathbf{B}(s_{i,j}) - (\mathbf{B}(\mathcal{M}_{i,j}) - p_{i,h_{post}}) |$$

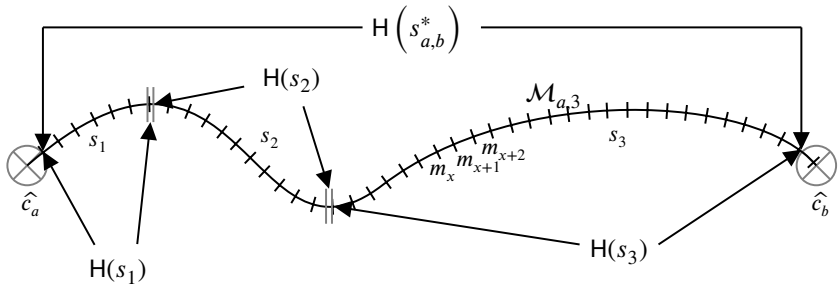


Figure 12.6 Illustration of difference between atomic heading changes and the ongoing curvature. Both information is used for the calculation of the heading change similarity metric and the segment sequence curvature similarity. The curvature is composed of single heading changes and describes the course of a road more closely.

where z is the total number of segments and l_i the number of segments between two turns \hat{c}_i and \hat{c}_{i+1} , i.e. $l_i = |s_{i,i+1}^*|$.

This approach allows scenarios such as the following to be considered accordingly. Given that two turns c_a, c_b are linked by $s_{a,b}^* = [s_1, s_2, s_3]$. Now, if $B(s_1) = -B(s_2)$ holds, the heading change similarity metric would depend substantially on s_3 and the curvy progression of $s_{a,b}^*$ is lost. Figure 12.6 shows the relationship graphically again. The course of the line corresponds to the route as in \mathcal{G} . Each line corresponds to a measurement m . Road segments s_1, s_2, s_3 are confined by turns in between that are used in this approximation point.

However, this metric is sensitive to temporary events such as overtaking or specific driving behaviors as explained in Section 12.3.2. It also depends on the accuracy of the distance, which sometimes degrades (c.f. Chapter 4). Nevertheless, the metric has proven to provide valuable insight into the overlap of a route candidate with a path.

Traffic light
placement
similarity

Further support for decision making lies in consideration of traffic light positions along a route candidate r^* . Idle times in the sensor data can be caused by a traffic light (c.f. Section 8.4). The goal of the metric is to describe how many of these idle times match traffic light positions in the map data. Note that the stationary phase does not necessarily have to occur precisely at the traffic light position (e.g. because several cars are waiting). This is taken into account by defining an

area around a traffic light as a detection zone. Hence, we define

$$\chi_{TL} = 1 - \frac{p}{q}$$

where q is the number of detected traffic lights and p the number of matching traffic lights. If no traffic lights are detected, set $\chi_{TL} = 0$.

Furthermore, traffic lights allow the meaningful exclusion of candidates on semantic terms. The construction of \mathcal{G} also included information about speed limits. An educated guess can be made based on both indicators: We assume that it is improbable to observe a long idle phase (due to a traffic light or not) on a road segment with a speed limit of 100 km h^{-1} or a specific type. Consequently, we may exclude such road candidates similar to Li et al. [231].

Evaluation

12.6

We now evaluate the proposed attack. First, we describe the setting and present the default parameters. Subsequently, the quality of the sensor data is examined, showing how the thresholds are defined based on the findings. The success rate of the attack is comprehensively investigated based on three areas and a comparison with related work, namely Narain et al. [276] Waltereit et al. [389] is shown. The filters and scoring mechanisms used in the attack are then considered to conclude their goodness. Furthermore, the influence of other parameters on the success rate is considered. Last, an analysis of the runtimes follows accordingly in order to be able to make a statement about the required efficiency.

Structure

To assess the ability of the approach to infer a correct route, we define a set of k elements that includes the top- k elements of a candidate list \mathcal{R}^* ordered by the score defined in Section 12.5.3. Observing a list of candidates enables us to measure an attacker's competence in reducing the search space to a specific level. The given position of the correct within the k elements is subsequently denoted as *rank*. Furthermore, we differentiate between an *exact match* and a *rough match*. An exact match is defined as the exact match of the driven route with the candidate route, while a rough match loosens this constraint only by requiring the candidate route to overlap at least 80% with the actual route. Even a rough match allows an attacker to derive movement patterns of a victim, such as frequent locations, and still is a severe privacy threat, in particular, when collecting data over a lasting period.

Rough
match

12.6.1 Setting

The attack was evaluated using the real-world data pool introduced in Section 5.2. We limited the data sets to routes from the city of Regensburg, Germany, and its surroundings. Therefore, the data set contains realistic routes that also take into account the diversity within the sensor readings to respect the challenges explained in Section 12.3.2. If needed, smartphone-to-vehicle alignment was performed using the approach depicted in Chapter 4 to allow recurring data orientation that is needed for the attack to work. This is particularly true for readings from the accelerometer, as the next section will show.

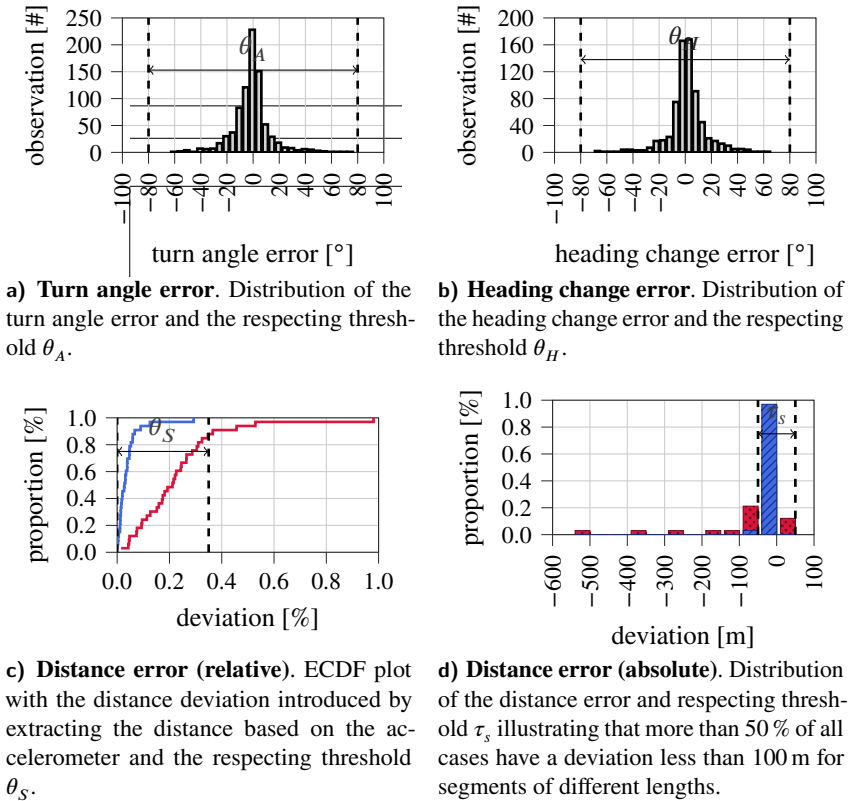
Table 12.3 Overview of default parameters that are used for the attack.

Parameter	Value
Scoring weights	$\lambda_A = 0.5$, $\lambda_D = 1.2$, $\lambda_H = 0.5$, $\lambda_C = 1.0$, $\lambda_{TL} = 0.2$
Distance error rate	$\theta_S = 0.35$
Road width threshold	$\tau_s = 50$ m
Magnetometer filtering threshold	$\theta_M = 90^\circ$
Gyroscope filtering thresholds	$\theta_A = 80^\circ$, $\theta_H = 80^\circ$
Slight turn threshold	$\theta_S = 60^\circ$

Parameters Several parameters are introduced to handle i.a. the difference between the reference data from OSM and the sensor data recorded. Also, parameters are used for scoring to select the best-fitting road candidate. The default parameters are shown in Table 12.3. Section 12.6.2 will explain the determination of threshold values. Also, we analyze the impact of altered parameter values in Section 12.6.4.

Benchmark with related works Narain et al. [276] provided real-world data sets for trajectories in the city of Boston and Waltham, USA. This data is used within the benchmark in Section 12.6.3 that compares the ability to infer trajectories between our approach and related work Narain et al. [276] and Waltereit et al. [389]. The data contained 69 trips in Boston and 63 trips in Waltham and were collected by four different drivers³. The provided data did not include all necessary sensor readings to derive the speed and, ultimately, distances based on sensor data. Hence we estimated the required properties using GPS traces.

³The routes provided were filtered accordingly by the authors: Some trajectories containing sensitive information have been removed, so there may be discrepancies between the evaluations of Narain et al. [276] and our replicated experiments.



Source	
---	DERIVED

Overview of the determination of different thresholds. The attack and respective thresholds are driven by sensor data (FPD) that differ in quality and accuracy. In particular, distance, turns and headings are of interest.

Figure 12.7

Sensor Accuracy

12.6.2

We recap the sensor accuracy that is relevant for the proposed attack to give an understanding of the limitations. As a side-channel attack, data acquisition is limited to zero-permission sensor data generated by the IMU. The fact that this data fluctuates qualitatively has already been discussed in the context of this work (see Section 2.3). Thus, corresponding thresholds were introduced,

Accuracy to define thresholds

which nevertheless allowed the execution of the attack. The thresholds will be quantitatively classified based on empirical data in this section.

Gyroscope Particular patterns in the sensor data allow the detection of traffic circles and curves. The procedure was described in Section 5.4 and Section 8.5. Essential for the detection of the given events is the gyroscope. In Figure 12.7a the deviation of the turn angle between the measured sensor data and the data stored in \mathcal{G} is depicted. This value is important to correctly select the corresponding candidates in \mathcal{G} for ps . Figure 12.7b indicates the strength of the heading change error, which ultimately enables the curvature calculation. The attack uses curvature to verify whether a selected road candidate is viable for a given path. The basis for evaluating the turn angle errors and heading change errors are the test routes and the corresponding curves and traffic circles that have been detected correctly. Even for 95 % of all values the error is smaller than 30° ; hence, the thresholds are chosen to include the whole dispersion including outliers, i.e. $\theta_H = 80^\circ$ and $\theta_H = 80^\circ$.

Accelerometer Distances are also recorded as part of the side-channel attack using the IMU, in particular the accelerometer. Deriving distances based on acceleration values is an error-prone process, especially in the case of arbitrary positions of the smartphone in the vehicle (see Chapter 4 for a detailed analysis of this topic). Distances, in turn, are an exclusion criterion when connecting pairs of curves and serve as the basis for calculating the curvature. To account for the inevitable deviation between the reference data (based on \mathcal{G}) and the measured data, we induced a relative distance threshold θ_S and an absolute distance threshold τ_S . Figure 12.7c gives an impression of the degree of degradation of distance accuracy of the distances derived from IMU and GPS compared to the reference road segments found in \mathcal{G} . We can state that even though the GPS distance has a lower deflection, the derived distances are still sufficiently accurate, as most of them are below our chosen threshold $\theta_S = 0.35$. In particular, 90 % of all road segments are up to 35 % shorter or longer compared to reality. The absolute numbers, as shown in Figure 12.7d, show that the major part of the deviations for derived distances via the accelerometer are in a window of up to ± 50 m. As expected, the GPS does better. The value of 50 m represents our absolute threshold τ_S .

Magnetometer The magnetometer is used to filter turns before finding pairs of turns in a successive step. It is used as a rough baseline to understand the direction of a vehicle when passing a turn. Hence, we chose a rather large value of $\theta_M = 90^\circ$. This is in line with values from the literature [276]. It is based fundamentally on the fact that, as already discussed, the magnetic field inside a vehicle is subject to



Overview of three different area sizes used for the evaluation. The geographical areas of different sizes are used to construct \mathcal{G} . The size impacts the complexity for trajectory inference.

Figure 12.8

disturbances, resulting in more significant deviations in the measurements. This was empirically verified by placing a smartphone at rest in various positions in the vehicle.

Success Rate

12.6.3

The attack will be evaluated using the Regensburg test region in the following. Thus, the correct detection rate of a driven route is of interest. At the beginning of this chapter and in accordance with the literature [134, 231], we motivated that the quality of recognition depends on the size of the region under investigation. This seems reasonable w.r.t. the number of potential routes and is also tested in this section. For this reason, we define three geographical areas Q1, Q2, and Q3 to investigate the recognition quality as a function of the region's size. All routes can be found in each search space to ensure comparability. The boundaries are shown in Figure 12.8 and are defined as follows.

Search spaces different in size

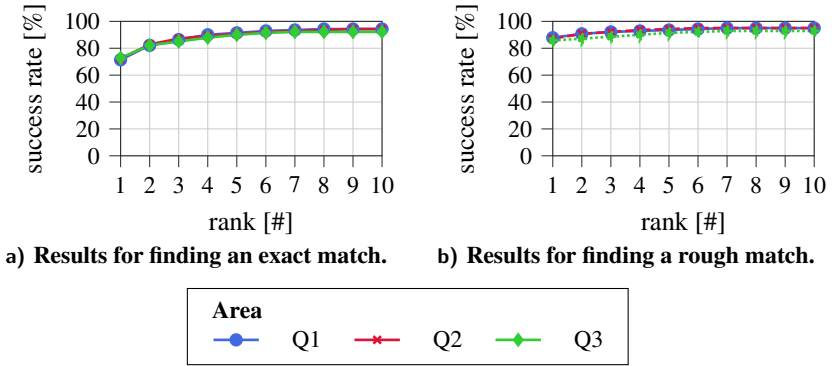


Figure 12.9 Success rate for the trajectory inference attack across all test areas Q1-3.

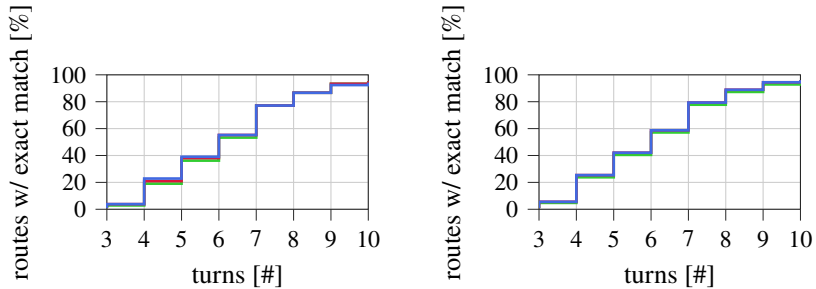
- Q1** An area ($\approx 400 \text{ km}^2$) covering the city of Regensburg and its surroundings to include all test routes.
- Q2** An area ($\approx 2000 \text{ km}^2$) covering the entire district of Regensburg including rural surroundings.
- Q3** An area ($\approx 4500 \text{ km}^2$) including Regensburg and Ingolstadt, a similar sized city.

First
impression

Figure 12.9 shows the success rate w.r.t. the three different geographical areas Q1, Q2, and Q3. Overall, we can see that the detection performance is similar for all three regions. From this point of view, the attack can be considered well-scaling as the recognition quality does not drop compared to other works [231]. In particular, the success rate of a perfect match w.r.t. the best-ranked route is 67.14 % for Q1, 66.43 % for Q2, and 71.43 % for Q3. However, while Q1 and Q2 have an almost identical success of 90.00 % (and 89.29 % respectively) when the five best-ranked routes are examined, Q3 decreases, as expected, to 87.14 %. It is noticeable that the success rate improves slightly for the largest region Q3, although more route candidates are formed. However, the correct routes can be determined more effectively from the sensor data due to better normalization combined with the established similarity scores. Eventually, the correct route achieves a higher score compared to other routes.

Impact of
the search
space size

It can be stated that between the consideration of the urban area of Regensburg (Q1) and the inclusion of more rural areas (Q2), a negligible difference in the recognition performance arises. One reason may be that other traffic structures are used in rural areas, which differ from the routes driven (recall that those routes are gathered in Q1). For Q3, peculiarly short routes or routes with only



a) **Top 1 candidate.** Distribution of finding the exact match as the top 1 candidate.

b) **Top 5 candidates.** Distribution of finding the exact match as the top 5 candidate.



ECDF plot illustrating the percentage of exact matches within different amounts of candidates in dependence of the number of turns. The area size has a negligible effect on the accuracy.

Figure 12.10

three turns were more challenging to find, resulting in a slightly lower success rate. The results for rough matching are similar to Q1 and Q2 but post slightly higher success rates than the exact matching case. In contrast, the detection rate drops slightly for Q3.

The proposed algorithm exploits turns to place path events p on a map in an initial step. The said turns, therefore, form the basis for detecting a path \mathcal{P} . In Figure 12.10, it can be seen that the correct recognition of the driven routes increases as the number of turns increases. This is also consistent with related work [231, 276], but the approach requires only a small number of such events for meaningful recognition. Figure 12.10a shows the relative number of exact matches where an attacker can trace the correct route uniquely. This is already possible with three turns to 70% and increases rapidly. Note that the median of turns within our data set is six turns. In Figure 12.10b a similar picture is drawn, where the correct route can be found within the top 5 candidates. For both cases, the influence of the area size is hardly decisive, which can most likely be attributed to the definition of the similarity metrics. In summary, the quality of recognition depends to a large extent on the turns, where the number of candidates, i.e. the uniqueness of a path event p and the turns to choose from \hat{C} are relevant.

Dependency on the number of turns

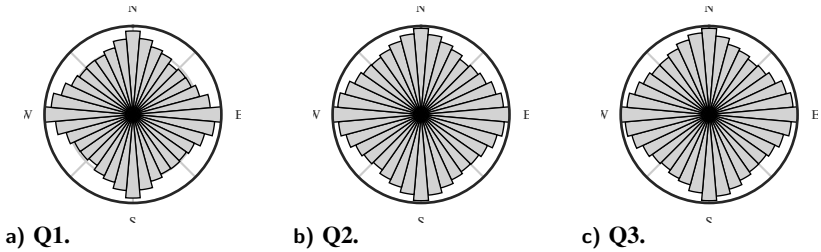


Figure 12.11 Polar plots depicting the heading distribution of the different areas. Each bin represents 10° . The areas are completely geographic oriented.

Impact of Turns

Therefore, in the following, we consider the heterogeneity of the road network based on Boeing [48]. The assumption is that road networks with turns in any cardinal directions (i.e. bearing), and angles (properties that serve as selection criteria) support high inference accuracy. We calculate the heading (or bearing) distribution of the underlying areas. They are shown as polar plots in Figures 12.11 a to 12.11 c where each bin represents 10° . It can be noted that although the north-south axis and the east-west axis outline the maximum, the road network is mainly balanced. Including a second city in the data set did not change the fundamental structure of the road network as reflected by Figure 12.11 c. This is different, for example, from grid-based cities such as New York City [48].

Since the street network is evenly balanced, a route may also be in any geographical direction, thus having a decent amount of entropy that may be used during turn selection, eventually reducing a set of candidates \hat{C}_i^* .

Comparison to Related Work

Notes on implementation

As introduced in Section 12.2, there are different approaches to the inference of the trajectory. We compare our attack with the approaches of Narain et al. [276] and Waltereit et al. [389] using both real-world data sets provided by the former work. This allows us to simultaneously evaluate our attack on a different road network to analyze any potential sensitivity related to different structures, e.g. turn frequencies. Looking at Figure 12.12, it can be seen that unlike Regensburg (c.f. Figure 12.12a), there is no north-south and east-west alignment, but the road layouts are slightly tilted (c.f. Figures 12.12b and 12.12c). At the same time, the network is sufficiently balanced so that sufficient preselection should be able to take place based on the turns. The data provided are trajectories in the cities of Waltham, and Boston, USA, collected by Narain et al. [276].

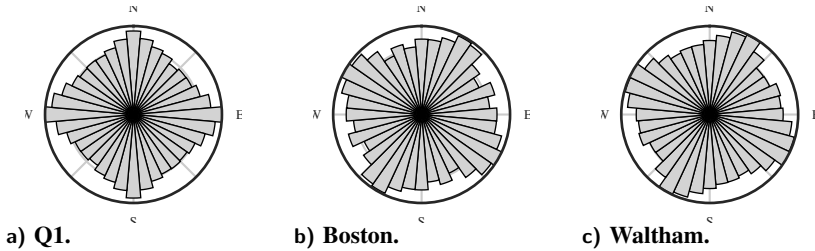


Figure 12.12

Polar plots depicting the heading (or bearing) distribution of the different areas used in evaluation. Each bin represents 10° .

We reran all experiments using data provided by Narain et al. [276], however, some recordings were excluded by the authors from the provided data set due to privacy-related issues, which eventually resulting in slight differences compared to the original work. Sensor data was converted accordingly to be feasible within our attack. Furthermore, we had to reimplement the attack of Waltereit et al. [389] to assess performance in the given data set. We set the allowed number of false positives and false negatives for their attack to zero, as we found that the attack performed much worse with the increased number of candidates caused by these tolerances when working with actual sensor data. Furthermore, the tolerance for distance error θ_S was increased from 0.15 to 0.3, which is consistent with the accuracy score of our equivalent parameter, based on sensor data. In fact, this should yield a more effective attack since more unreliable sensor data is considered. We also examine the success rate for trajectory inference in Boston and Waltham, since Waltereit et al. [389] did not include actual sensor data but synthetic data in the evaluation.

Adjustments
to related
work

The success rate of exact matches for both cities using the three attack approaches is depicted in Figure 12.13. Our proposed attack achieves a success rate of 50.72% in Boston and 70.96% in Waltham, considering that an exact match is to be found, i.e. the actual route must be the top1 candidate. Both attacks from Narain et al. [276] and Waltereit et al. [389] are inferior with 15.94% and 17.39%, respectively, in Boston, as well as 44.61% and 67.21%, respectively, in Waltham. The success rate for our attack slightly increases to 66.67% in Boston and 83.87% when extending the restriction to a top 5 candidate match. Similar behavior can be observed for the other two approaches as well, but they perform consistently worse. Recall that the orientations of the street network of both cities are similar to those of Regensburg. As expected, the results for Waltham are comparable to those obtained in Regensburg. In contrast, the recognition quality in Boston drops sharply, resulting in only two-thirds of all routes being in the top 5 (Regensburg nine out of ten routes). According to Narain et al.

Results
proving
superiority

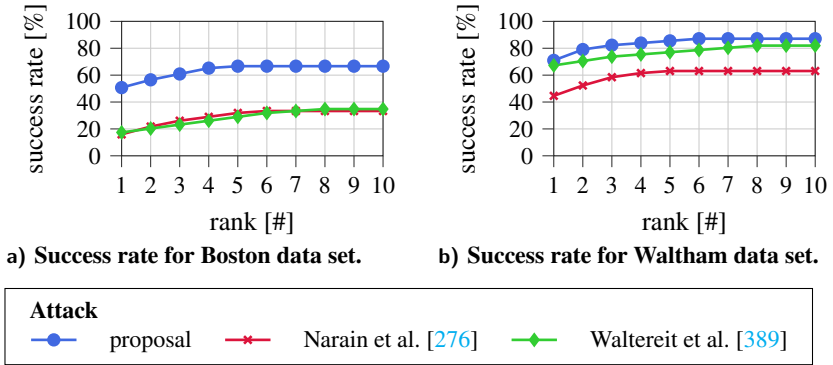


Figure 12.13 Comparison of success rate for an exact match with related works. Related works include the attacks of Narain et al. [276] and Waltereit et al. [389] and are benchmarked against the proposal. Data sets are provided by Narain et al. [276].

[276], there are various explanations for the poorer performance, such as heavy traffic compared to Waltham. We also observed traffic in our data set eventually degrading the recorded sensor data expressiveness, but Regensburg is relatively small compared to Boston. This is also underlined because unrecognized routes were not candidates in the top 10 for various reasons, which also holds for Boston and Waltham.

Reasons for
accuracy
degradation

Additionally, Narain et al. [276] acquired data before 2016, implying a poorer quality of smartphone sensors for data collection than the data gathered in our data set. We noticed that substantially more routes could not be discovered within the graph for Boston than for Regensburg, despite having more than twice the number of routes accessible for Regensburg. This may prove inferior sensor quality since the entropy of the road network is similar to Regensburg w.r.t. Figure 12.12a and Figure 12.12c. Hence, similar results were expected but are not reflected mainly in Boston.

12.6.4 Impact of Filtering, Ranking, and Parameters

Different methods are applied to reduce the number of turns in the turn candidate sets \hat{C}^* to improve efficiency. Furthermore, route candidates $r^* \in \mathcal{R}^*$ are ranked based on their similarity to \mathcal{P} . Lastly, different thresholds are introduced to handle inaccuracies in sensor data but also OSM data. We now look at each of them in detail in the example of Q1.

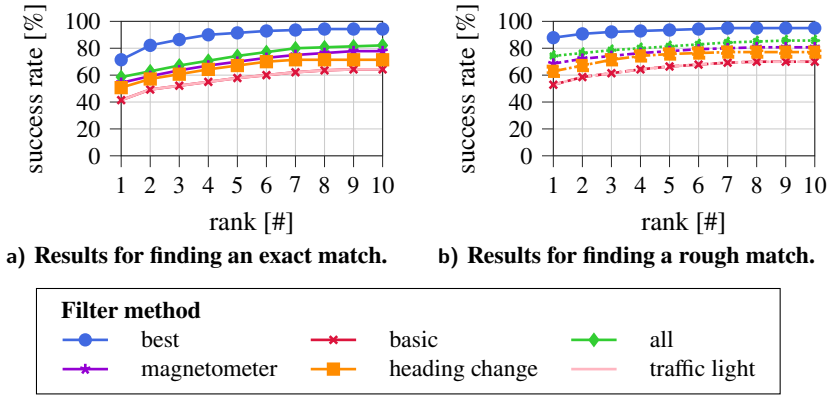


Figure 12.14

Success rate for the trajectory inference attack across all test areas Q1-3.

Filtering

Filtering is used to enable efficient computation and a manageable number of candidates. Filtering was selectively enabled to assess the performance of each filter. The results of the success rate are shown in Figure 12.14. “Best” represents the final approach, including filtering and ranking as previously evaluated. “Basic” represents a baseline similar to the initial proposal of Waltereit et al. [389]. “All” is a combination of all filters but without further optimizations. Three filtering approaches are present: traffic lights, expected heading change induced by road curvature, and orientations after turns based on the magnetometer.

In both cases, the search for the exact match and the search for a rough match, the filtering based on the magnetometer and the heading change is suitable. Both increase the success rate by about 10 % - 15 %, whereas the distance between both filters increases with the extension of the considered range (top 1 to top 10 candidate sets). The improvement is less evident for filtering road segments based on recognized traffic light patterns and the corresponding exclusion of unrealistic occurrences. This filter also improves the recognition compared to the basic method but is correspondingly expensive in terms of runtime. The combination of all filters is slightly above the course of the respective single filters of magnetometer and heading change and 17 % above the basic method. Consequently, both do not increase the detection rate in total but only proportionally.

Impact on success rate

Filtering is used not only to increase the success rate but also to significantly decrease the attack runtime, with the magnetometer filter reducing the runtime

Impact on runtime

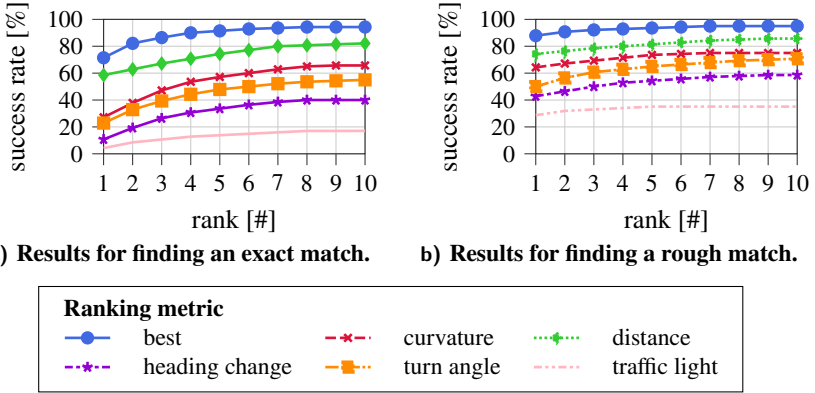


Figure 12.15 Success rate for the trajectory inference attack across all test areas Q1-3.

about 65 % and the heading change filter reducing it to around 38 %. However, including the traffic light filter slightly increases the runtime. In particular, using all three filters results in a reduction in runtime by about 71 % since the number of turn-pair candidates and hence the size of the Cartesian product is considerably smaller.

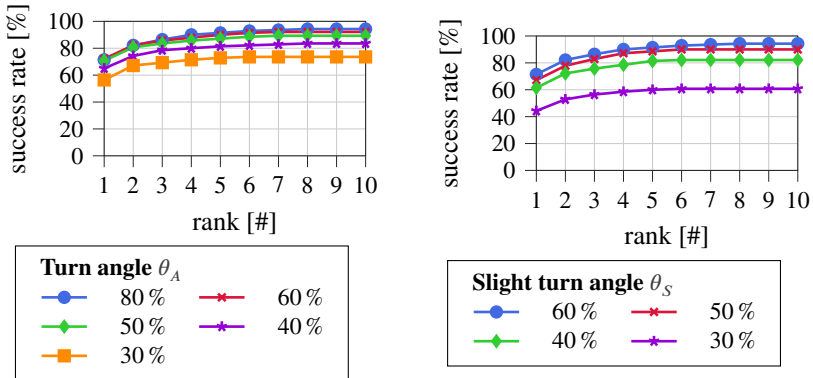
Ranking

Depending on the street network \mathcal{G} , there can be a large number of candidate routes. For example, when trying to infer routes in the largest search space, Q3, a route had 185146 different candidate routes selected based on the filtering methods. However, because of the sophisticated ranking, the correct route could be found as an exact match in the top 1 set. Setting the weights λ to zero, except for the metric of interest, allows analyzing the impact of each metric on the success rate⁴.

Scoring to support candidate selection

The consideration of the impact of a metric on the ranking of a candidate and ultimately supporting decision-making can be seen in Figure 12.15. Subsequently, the distance-based metric is shown to be very well suited to map a candidate to the route appropriately well, although it should be noted that this also requires a correct distance calculation in the sensor data. Furthermore, the inclusion of the curvature metric shows similarly promising results, so the newly proposed metric is a valuable extension for problem-solving compared to existing metrics such as the turn angle or heading change. The least detrimental effect on the

⁴The traffic light metric was computed only for routes that did contain the given and detected element.



a) Impact of different turn angles thresholds θ_A on the success rate. b) Impact of different slight turn angles thresholds θ_S on the success rate.

Impact of different settings for thresholds applied during database creation when identifying turns. Figure 12.16

ranking is determined by traffic light similarity due to comparing detected traffic lights with potential traffic light positions. However, this metric cannot distinguish between similar routes that pass all traffic lights and, for example, only differ in the part of the route where no traffic light is present. This assumption is underlined by the fact that the metric performs better in percentage terms in the comparison when searching for a rough match.

Thresholds

For the sake of completeness, we also want to discuss specific thresholds that were incorporated into the attack to handle unknown environments with changing traffic circumstances and low-quality sensor data. Therefore, we analyze the impact of the turn angle threshold θ_A and the slight turn angle threshold θ_S . The distance threshold θ_S was determined on the basis of the sensor accuracy in Section 12.6.2.

The turn angle threshold θ_A defines the point at which $e \in E$ are considered a turn and thus are included as an element in \hat{C} . The set is searched during the attack using path events p_i to identify potential candidates. Consequently, with more elements considered as turns, more elements are available to choose from for this selection process. The probability of picking up the right turn increases, but at the same time, the set of candidates also increases with lower θ_A . Figure 12.16a

Turn angle threshold θ_A

provides information about what effect the parameter has on the detection rate. A conservative value of 80 % was assumed for the attack since the risk of missing a turn is minimized due to the modeling of the slight turns. However, $\theta_A = 60\%$ has a decent success rate. Lower thresholds decrease the success rate even more since more turns are present in \hat{C} , although they do not reflect the corresponding driving maneuver in the real-world sensor data.

Slight turn
angle
threshold θ_S

The second parameter called θ_S takes into account the challenging task of differentiating a turn from a curvy road. Hence, such events are considered either a turn or a straight. Eventually, missing a slight turn does not result in a route not being able to be matched, but the creation of multiple route candidates based on either a turn or a straight. Based on the results depicted in Figure 12.16b, we can state that a threshold of 30° and 40° is too low to grasp slight turns. Using higher threshold values enables the algorithm to handle inaccurate sensor data adequately and still achieve high success rates at the expense of the runtime. Filtering, however, was able to reduce the number of routes even for large values of θ_S .

12.6.5 Feasibility

Python and the Pandas library were used to implement the presented attack. All tests were run as unconstrained Kubernetes workloads on an Ubuntu 20.04.1 host with an Intel Xeon CPU E5-2680 v3 (24 threads and 12 cores) running at 2.5 GHz and 64 GB of RAM. Sensor data are pre-processed in 2.47 s on average, with a standard deviation of 0.67 s. The time required to find a route depends on the size of the area, the number of turns, the uniqueness of the route, and the distance. The average time required to find a route within the smallest area Q1 is 1.02 min, with a standard deviation of 0.48 min. The average time required to find a route within region Q2 is 2.71 min, with a standard deviation of 1.26 min. Attacking the largest area, Q3, takes an average of 11.60 min, with a standard deviation of 9.97 min. Although this is not a formal test, the results show that the attack is possible even against opponents with modest resources. Significantly tighter filter thresholds can minimize runtime in exchange for lower attack success.

12.7 Conclusions

Trajectory inference is another class of attack that is present in the UBI context due to the transmission of sensitive sensor data. Based on the literature and our

own attack, we have shown that there may be a misconception on the part of the users between the nature of the transmitted data and its meaningfulness.

Overview of Literature

12.7.1

A literature review of work on route inference in the context of smartphones has shown that few but various approaches exist for the given task of reconstructing or even interpolating a trajectory. Such methods allow users to be tracked based on sensor data, including zero-permission sensor data, and thus significantly violate their privacy. In particular, most approaches do not define themselves as an attack but rather as an approach to reduce or replace the usage of traditional orientation methods such as GPS. The publication format supports this argument because most of the work did not appear in privacy-related formats.

Summary

We found that most works require a rough estimation of the area of a user to be feasible and efficient apart from the ones that require learning data in the first place. Such attacks are not applicable in the context of PHYD. It can be concluded that the more sensors used, the larger the area that can be analyzed, although not all work was specific concerning this topic and did not provide a thorough evaluation. One can argue that the combination of multiple attacks may even reduce the constraint of known areas; for instance, combining pressure-based approaches with sophisticated route estimation methods using IMU data may result in more severe adversary models.

Constraints

Our Approach

12.7.2

Based on the literature findings, we present a highly scalable attack designed, at the same time, to be robust against errors in the sensor data. It integrates and extends features of different approaches into a holistic concept; especially curves, heading, and distances, as well as curvature, are included in the algorithm along with newly proposed features such as traffic circles. Using the proposed algorithm, an attacker can find a route to get to an area without knowing the start and end positions. Unlike some related work, it is specifically tailored to the scenario of zero-permission sensor data and provides insight into the privacy threat. In addition to the topic presented in Chapter 11, a detailed motion map of a user can be collected in combination with a trajectory attack.

The approach is based on detecting directional changes in the sensor data, taking into account discrepancies and uncertainties in the sensor data and the reference map material accordingly. Specifically, the uncertainty of turns in sensor data and their modeling in OSM is adequately addressed by the concept

Evaluation

of *slight turns*. The diversity of the driving distances and the final goodness of the evaluation are ensured by various own and external data sets. For an area of 4500 km² the presented method manages to detect a route in 72.86 % of all cases, whereas the detection quality depends on the available features. For example, the number of curves and the uniqueness (i.e. curvature) of road segments as connections between curves are crucial. If the attacker pursues the goal of limiting the candidate routes and considers the five most likely routes, the probability increases to 88.57 %. Compared to related work, the recognition score for external data sets is up to a factor of 2.5 higher than the competitor results, showing good scalability and independence from smartphone and driving behavior.

Sources of error Analyzing the routes that were not found by the proposed algorithm yields two reasons, either related to uncertainty in event detection or low-modeled map data. Both cases result in a discrepancy in mapping one to the other or vice versa.

Uncertainty in Event Detection Sensor data is converted to a path \mathcal{P} that contains path events p that are eventually mapped to elements in the street network. Hence, it is assumed that p are present in the map data, although there may be occasions where that is not the case. For example, we found that turns—even though only slight turns—are implicated through external factors such as road works that may require a temporary detour yielding a false-positive turn in the sensor data. Furthermore, particularly broad and curvy roads are hard to detect because of a specific balancing of respective threshold parameters to achieve an overall high success rate that could not account for the varying number of detected turns on such roads across drivers. Even though specific features, namely traffic circles, are of high information due to their comparatively low number of occurrences in street networks and hence essential during candidate reduction, they may be false friends. We found that in a single instance, a driver processed a sequence of right and immediate left turns similar to a traffic circle pattern (see Section 8.5), eventually eliminating the correct candidate from the set of candidates. This stresses the fact that balancing different driving behaviors is challenging.

Inaccurate Map Data Features in sensor data can also not be found in the map material if they erroneously do not exist or are not modeled accordingly. The former is caused by missing tags on elements (e.g. missing the tag representing a traffic circle). The latter is a combination of driving behavior and oversimplification of street structures which can often be attributed to the lack of road width modeling or the inability to represent natural driving patterns. Therefore, we stress that high-quality map material is required that adequately models the real world.

We now want to address some limitations of the attack. First, the attack requires velocities based on the accelerometer. Hence, the requirements and constraints explained in Chapter 4 are applicable. In the given scenario, a smartphone is considered to gather the sensor data required to record the trajectories. However, the evaluation assumes that each \mathcal{M} yields a single trip. An adversary likely collects a continuous stream of zero-permission sensor data. Hence, detecting the start and end position of a trip is essential to perform trajectory inference *ex post*. The applied street network \mathcal{G} contains information available in OSM limited to public roads, eventually excluding gated infrastructures such as private roads as they are found on private properties. Furthermore, we excluded parking spaces, as they are generally quite similar and do not provide much information. However, especially such structures yield turns that will be picked up by the attack but will eventually not match in \mathcal{G} . Such situations have to be detected and removed during a preprocessing step, for example, by using low speed and frequent turns as indicators. Lastly, a candidate and eventually the best guess is a sequence of road segments and road connections (turns) delimiting the route. The attack cannot find the exact starting or ending position if it does not coincide with a turn because previous sensor readings and movements are ignored at the time of the creation of \mathcal{P} . Still, this yields enough knowledge for an adversary to track a user to his points of interest. Creating relative movements similar to [163] with the road connection as a starting or endpoint, respectively, can be used to derive or estimate more precise locations, such as parking positions on the road or parking lot positions.

Limitations

The attack significantly extends existing approaches to trajectory inference by supporting, analyzing, and evaluating various features. In addition to curves, street curvature, heading changes, and features of the two-dimensional space are included. An augmentation around altitude similar to [172] is comparable to curvature but could support the selection of route candidates. As described in the limitations, detecting curves is difficult due to various factors, such as driving behavior or route characteristics. The approach should be extended in the sense that it can account for incorrectly detected curves accordingly, such as in [389]. This would make the recognition more robust. Furthermore, it is of interest to what extent grid-like structures, such as Manhattan in New York, affect the recognition. In the exemplary area, the similarity of the curves and the length of the route segments are very high, which could cause our approach's two essential selection and evaluation criteria to lose efficiency.

Outlook

12.7.3 Privacy Implications

The fundamental right to privacy of users is also severely limited by location inference attacks and thus contradicts the rights of users established by Solove [351]. Accurate identification of trajectories based on sensor data has far-reaching implications for users. The attacker model in the context of PHYD is greatly enhanced by the side-channel attack (we here ignore the fact that the offending GNSS data is a privacy concern and yet is transmitted).

Limit on power We cannot speak of *limit on power* when transmitting raw sensor data. A data processor not only gets the possibility to evaluate the driving profile of a user but also to infer his whereabouts. Such capabilities were basically only available to Location-based Service (LBS) providers, as users there typically transmitted location data. Similarly, this can apply to mobile network providers, as these could also create extensive movement profiles based on technical necessities. Now, this possibility is transferred to another service provider, who does not need to track exact locations. Even more severe is that our attack only uses zero-permission sensors that can be gathered without user knowledge in the background within any other decoy application. Consequently, every application developer is upgraded to a LBS provider and gains capabilities that were previously only available to highly scrutinized services (for example, cellular providers). In the context of PAYD, it is sufficient to know the approximate location to be able to make a rating based on city/non-city, for example.

Appropriate social boundaries With precise movement profiles [132], an adversary can gain deep insights into a person's life. *Appropriate social boundaries* can no longer be guaranteed. For example, if a provider can see that a person regularly travels to a hospital, inferences can be drawn that go far beyond the boundaries of a UBI. It is a profound violation of personal liberty. In addition, a person can be discriminated against based on his visited places, eventually reflecting his preferences. As UBI implicitly includes the location within PAYD, it is unknown how this information may be included in the premium calculation.

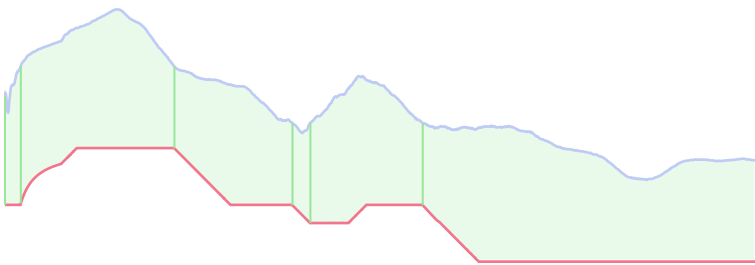
Reputation management Whereabouts per se are critical in terms of personal *reputation management*, as they allow insight into the daily routine without a person actively communicating it. In terms of reputation management, an individual should be allowed to move anonymously without revealing specific locations such as hospitals or workplaces. With the concatenation of external data sources, unique movement patterns can be established based on travel distances, times, and in combination with the previous driver identification attack (c.f. Chapter 11).

This brief discussion is intended to show that effective side-channel attacks enable extensive privacy violations. These can only be prevented by modifying the data appropriately before it leaves the user domain so that the attacks, as mentioned earlier, can no longer be used in a meaningful way. This also applies outside the UBI scenario. At least attack countermeasures can be applied. In the context of a PET such as kubi (c.f. Chapter 14), the sensor data can be inverted, for example, so that the attack should no longer be possible since the corresponding inferred curves no longer correspond to the real driving route. However, the quality of local services can still be guaranteed with local knowledge of the data manipulation process.

Countermeasures

Part IV

Protection of Privacy



Overview of Privacy Enhancing Technologies in Sensor-Focused Environments 13

Driver identification and trajectory reconstruction, as illustrated, pose a severe threat to privacy and ultimately undermine the individual’s right to informational self-determination. To protect this fundamental right, the GDPR establishes six principles for data processing, already introduced in Sections 1.2 and 9.4. The information disclosure process is a complex balancing of a user’s interests consisting of pragmatic and subjective decisions [407]. Two elements of GDPR are of particular interest in this context, namely *data minimization* and *transparency*. Consequently, a user wants to disclose only the minimum amount of information needed for a particular task while being optimally informed about how data is or has been used. Technology should assist the user in this process and provide convenience [255].

“[T]he legal privacy principle of data minimization by minimizing or avoiding the collection and use of personal data of users or data subjects” [129] can be enforced with the help of a PET. PETs go back to John Borking [315] and address the anonymity of a user, the unobservability of his acting, and the unlinkability of successive events, with all of them being confidentiality aspects [126]. With TETs, a second concept exists that addresses “*informed consent and transparency*” [129]. The focus is different compared to PETs, yet both of them aim to increase a user’s privacy. TETs can further be separated into *ex-ante* and *ex-post* transparency, with the former addressing the indented data processing and the latter addressing the actual data processing [129]. This can be projected on the information disclosure process of a user that also

Protecting user interests

Acknowledgement

Parts of the research presented in this chapter are based on supervised work [S6, S14].

differs in the intended and actual behavior (i.e. the privacy paradox) and shows how such tools can help a user make an informed decision. Although overlaps between both privacy-enhancing technologies are present, in particular, in terms of transparency¹, a distinction between both concepts is common (c.f. [170]).

Privacy by
design and
PETs

A third construct closely related to PETs is Privacy by Design, yet they differ fundamentally. PETs are mainly used *ex-post* to protect user-preserved data and are considered agnostic to a specific service or product. Depending on the protection techniques used in aPET, the functionality of the actual application is limited in properties, not limited to quality, functionality, or availability. Thus, it can be concluded that there is inevitably a trade-off between the protection of personal interests and the merits of functionality or application. Privacy by Design addresses this false dichotomy and defines that systems should be designed to provide all functionalities while ensuring privacy and providing personal control over one's information [66]. Accordingly, such solutions are also of interest because they provide immediate and sustainable benefits to users. For a closer look at Privacy by Design, refer to Section 6.1.

Contribution

This chapter is intended to provide an overview of current approaches to enable privacy for use cases processing sensor data from smartphones w.r.t. this work. Therefore, a SLR is executed to provide

- ▶ an overview of current PETs and their respective architecture in the mobile ecosystem and
- ▶ a technical analysis of PETs in terms of the protective measures applied.

Structure

Section 13.1 starts by introducing the research questions and the document corpus. Next, Section 13.2 analyzes and discusses the identified approaches and gives a structured overview. The chapter is concluded in Section 13.3.

13.1 Structured Literature Review

We now introduce our research questions that are subject to be answered within this survey. Then we describe the search process and finish with a high-level presentation of the relevant finding.

¹As we will see in this chapter, PETs often offer a monitoring functionality along with other active privacy-preserving methods. Monitoring, however, is considered to provide transparency which, by definition, is the purpose of an *ex-post* TET.

Research Questions

13.1.1

The subject of the SLR is to understand “*the design of mobile-first Privacy Enhancing Technologies for sensor data?*”. This includes understanding the design of PETs in the mobile scenario when enabling the support for sensor data processing. Furthermore, the applied techniques are of interest because, as we have already learned, simple perturbation or suppression techniques are not feasible in particular in the UBI context [322]. Therefore, we ask the following RQs to gain insight into the given topic.

- Q1** What are design principles when it comes to PETs for sensor data in smartphone scenarios? How is the user involved in this process?
- Q2** What building blocks enable privacy w.r.t. the sensor data produced in the given context? How adaptable are the methods used?

Search Process

13.1.2

The search process for the SLR is based on Kitchenham and Charters [209].

The search focuses on ACM Digital Library, IEEE Xplore, ScienceDirect, and SpringerLink as a resource for publications. We iteratively developed a search term² that is feasible to identify the relevant work needed to answer our questions.

```
( TET | PET | PRIVACY ENHANCING | PRIVACY PRESERVING )
& ( SMARTPHONE | ANDROID | IOS )
& SENSOR
```

We deliberately decided to include TETs as a search term in order not to miss any essential works. Works that do not adequately present and address PETs were subsequently excluded i.a. based on the inclusion and exclusion criteria. The criteria are as follows:

1. First, the work is presenting and explaining in sufficient technical depth a PET, ultimately excluding survey papers.
2. Next, the scenario of sensor data generated by mobile devices limited to smartphones is focused on.
3. The proposal is based on existing structures of sensors and their application in existing mobile OSs; instead, data is generated analogous to the description of this work (see Section 5.2).

²For an explanation of the notation, see Appendix B

4. Works with a too-narrow focus on eHealth and Smart Cities are not considered.

In addition, only peer-reviewed work in English was considered that was available to the authors. It is assumed that the peer review process provides only high-quality content with comprehensible attacks, although the quality was again assessed during the SLR. As in the previous SLRs presented in this work, we also perform a backward search in addition to the forward search. For instance, we include work that may be of high relevance based on citations.

13.1.3 Relevant Findings

Table 13.1 Overview of the 33 publications identified in the SLR. The works are assigned to different disciplines.

Publication	Year	Publisher	Field
Zhang et al. [422]	2020	Elsevier	Security and Privacy
Han et al. [165]	2020	IEEE	Security and Privacy
Wu et al. [405]	2019	IEEE	Network
Miao et al. [264]	2019	ACM	Systems Design
Luo et al. [238]	2019	IEEE	Network
Shen et al. [340]	2018	Elsevier	Network
Romero-Tris and Megías [314]	2018	ACM	Knowledge Discovery
Mirzamohammadi and Sani [266]	2018	IEEE	Mobile Computing
Malekzadeh et al. [242]	2018	ACM	Security and Privacy
Jourdan et al. [194]	2018	ACM	Mobile Computing
Han et al. [164]	2018	Elsevier	Mobile Computing
Lyu et al. [240]	2017	ACM	Knowledge Discovery
Krupp et al. [218]	2017	IEEE	Mobile Computing
Huning et al. [183]	2017	ACM	Mobile Computing
Bai et al. [31]	2017	Springer	Security and Privacy
Saleheen et al. [330]	2016	ACM	Mobile Computing
Mense et al. [263]	2016	ACM	Mobile Computing

continued on next page

Luo et al. [239]	2016	ACM	Mobile Computing
Zhao et al. [424]	2015	IEEE	Mobile Computing
Xu and Zhu [410]	2015	ACM	Security and Privacy
Dang and Chang [92]	2015	ACM	Security and Privacy
Supriyo Chakraborty et al. [360]	2014	USENIX	Security and Privacy
Rasthofer et al. [308]	2014	IEEE	Security and Privacy
Cappos et al. [62]	2014	IEEE	Mobile Computing
Biswas and Vidyasankar [46]	2014	Springer	Computer Science
Chakraborty et al. [68]	2013	ACM	Mobile Computing
Raghavan et al. [306]	2012	ACM	Mobile Computing
Götz et al. [147]	2012	ACM	Systems Design
Ghosh et al. [145]	2012	IEEE	Security and Privacy
Dua et al. [116]	2012	Springer	Security and Privacy
Canlar et al. [61]	2012	Springer	Security and Privacy
Cristofaro and Soriente [87]	2011	ACM	Security and Privacy
Choi et al. [81]	2011	Springer	Security and Privacy

After performing a tool-assisted selection process, 33 works were identified as relevant w.r.t. the applied inclusion and exclusion criteria. All results are listed in Table 13.1. Most publications were found based on the forward search, and three were included in the backward process. One can see an almost even distribution year-wise, with a peak of literature coming up in 2018. The origin is dominated by events and journals from the field of security and privacy (39%) as well as mobile computing (36%). Mobile computing here also includes sensor-focused events. Furthermore, half of the literature was published by ACM, followed by IEEE and Springer. Subsequently, we extracted relevant attributes from the literature to answer the RQs in the next section.

Results

13.2

We now take a look at the design principles and the building blocks to answer the questions posed.



Figure 13.1 **Proposals in the literature can be roughly divided according to the implementation approach.** On the one hand, there are solutions that function independently, and on the other hand, solutions in which Privacy by Design was considered during the development of the use case. In the corpus of documents, the former class slightly predominates.

13.2.1 Design Principles (RQ1)

In order to answer the first question about the design principles, the various works are first examined w.r.t. the form of integration. It is noticeable that the works can be differentiated based on their implementation. Either they deal with on-top solutions for the protection of the user's privacy for third-party systems or that the application or the system itself was already designed under privacy-friendly conditions. In the latter case, we are dealing with privacy-by-design approaches. This approach was also followed in Part II in the context of this work.

Types of solutions

Looking at Figure 13.1, an almost balanced picture is presented. Eighteen papers are dedicated systems that comply with the definition of a PET, and 15 papers are architectures that follow the Privacy by Design approach. Especially crowdsensing applications are designed according to Privacy by Design guidelines. Most of the time, they discuss the question of to what extent the interests of the various participants can be taken into account. For example, systems can be found that design evaluations based on sensor data from users with appropriate security and integrity [183, 264, 422]. More generally, for example, Luo et al. [238] interpret the problem and presents a system that can modify sensor data accordingly based on differential privacy and works independently of a specific application. Consequently, it can be considered a building block for other applications. Representatives of the PET class are deployed solutions that work independently of applications and often provide monitoring aspects [31, 62, 165, 218, 266]. Furthermore, there exist nevertheless PETs designed for a

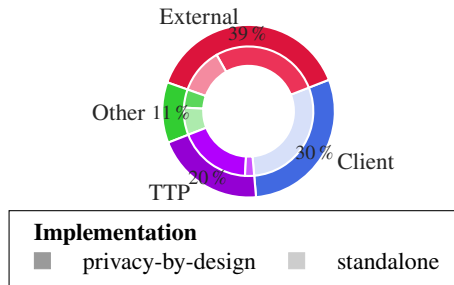


Figure 13.2

Overview of different requirements that the different approaches pose. Most of the works rely on an external infrastructure to provide privacy to users. Less common are proposals that are running within the user domain. Also some publications introduce a TTP that ensures a specific level of privacy.

specific group of applications and so, for example, applicable only for LBSs scenarios [164].

This leads to the next inquiry. We also analyze the underlying structure of the privacy concept and how privacy is brought into a domain. For this purpose, Figure 13.2 compares the previously considered implementation type with the requirements that must be met for the protection solution to be realized. Three general artifacts can be distinguished. 39 % of the identified work uses an external infrastructure to process user data. PET solutions are rarely seen in this category. Mense et al. [263] present a PET that uses an external server to provide privacy as the information is routed through this external infrastructure that can then perform further protective measures. Similarly, the approach of Choi et al. [81] stores data in a dedicated database that is then accessed through a broker that eventually enforces a user-defined policy. Hence, both solutions replace direct access to a resource to access it via a proxy. In systems developed from the outset with privacy in mind, data is often processed remotely only data that is appropriately non-critical [46, 61, 87, 92, 183, 238, 240, 264, 314, 340, 405, 422]. Exclusively PETs are found in the category of client-side solutions [31, 81, 145, 147, 165, 218, 263, 266, 306, 308, 360, 410, 424]. Such retrofitted applications usually work as interceptors. They check and control the flow of information between the mobile OS and sensors towards an app. Bai et al. [31], for example, replaces API calls to sensors and returns data to an application in a privacy-friendly manner. In addition, a third class is found to be TTP-based solutions, which are exclusive to Privacy by Design concepts [61, 87, 183, 242, 264, 314, 405, 422]. There is overlap here with the external server class, for example, when

Architecture

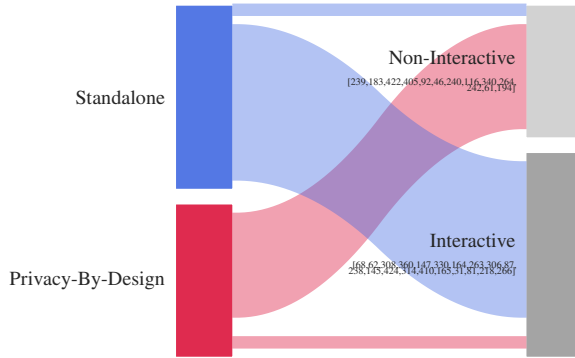


Figure 13.3 Overview of the interactivity of methods found in the literature. Methods are either require the user to interact with the protection algorithm or are working in a transparent way without the need for further adjustments or decisions.

data is stored remotely, as in the case of Choi et al. [81], but made available to other participants in anonymized form through an TTP. Malekzadeh et al. [242] is exclusive to this class since a pre-trained model must be made available locally accordingly as part of the proposal. The distributor is interpreted as TTP in this consideration. Other, as a catchall class, includes work [68, 116, 164, 194, 239] that does not fit into the other three classes due to e.g. shifted focus. For example, Dua et al. [116] is a proposal of a proof protocol to convince a user that data can be shared. It is a building block for a privacy-friendly solution but not a viable design in its own right.

Architecture
and TTP

There are some critics of relying on a TTP. TTPs conflict with the fundamental requirements of PETs such as “no trust into the network operator” and “no trust into one centralized station” [126]. Consequently, a strong PET or even a Privacy by Design architecture ultimately avoids the need for a trust anchor by design. That such a design is possible is shown, for example, by DC networks [73] (for communication) or blind signatures [72] (for privacy-preserving transactions).

Interactive-
ness

Figure 13.3 shows a strong correlation between the type of implementation and the level of interactivity. This is prevalent as Privacy by Design architectures are built with privacy, trust, and integrity in mind. Non-interactive approaches are preferable w.r.t. bounded rationality, as they often relieve the user of making complex decisions that are difficult to assess. Some PETs, although listed as interactive, are driven primarily by a policy that can be adjusted (and, therefore,

are interactive). However, the user itself is not necessarily required to create or maintain such a policy.

To fit into the mobile context of this work with UBI as a running example, we focus on two stakeholders, namely the data processor and an application provider. We deliberately exclude the user from the stakeholders analyzed within this chapter because we argue that privacy and, therefore, the application of a PET is in his interest. A data processor is a data-consuming entity in this context, i.e. he uses the information provided by users to e.g. analyze, process, aggregate, or distribute it accordingly depending on the use case. However, the application provider is the entity that develops and provides the application to a user but does not receive it. For example, an email client may be developed by Google, but emails are sent to another party. A PET should protect in the best case against both parties. However, this is not the case. We found that 76 % of the identified works protect against an application provider and 61 % against a data processor. When this is broken down to the implementation, it shows that 30 % of the Privacy by Design platforms protect against the application provider, i.e. the one providing the platform. With 42 %, protection is provided against the data processor. For PETs, the application provider is 2.5 times more common than the data processor with 45 %. This may be due to the alignment of PETs, which often runs locally on a smartphone and generically observe the sensor data consumption of third-party applications. In 45 % of all cases, both stakeholders are considered.

Adversary

Building Blocks (RQ2)

13.2.2

We now have learned how PETs are designed from a system architecture perspective. Next, the data alteration process is focused on, as it is an essential component of a privacy-enhancing method. Due to the context of sensor data and with business models like UBI in mind, we will also take a detailed look at each protective measure.

In the next step, sensor data is assigned to corresponding classes that correspond to a rough use case and can be applied to the side-channel attack classification in Chapter 10. Figure 13.4 shows the distribution according to the implementation. On the one hand, these are location-specific data such as the position of a user, vehicle, or device (stationary or moving). On the other hand, environmental sensor data includes temperature, brightness, and others. Furthermore, other sensor data is rarely addressed in the document corpus, such as the audio (i.e. microphone) [340]. In addition, a generic class was included. Such approaches are not specifically tailored to protect against a predefined set of data but can work

Types of protected data

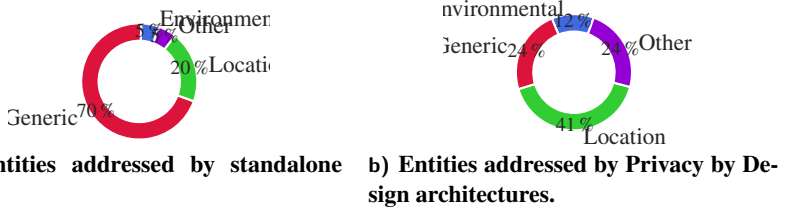


Figure 13.4 Distribution of the addressed entities that are to be protected by the privacy-enhancing technology.

in a use case-agnostic way. It is obvious from Figure 13.4a that PETs agnostic in terms of data type; hence, they can be used for a multitude of use cases. This is a clear advantage, as there is no need for application-specific solutions to enable privacy. However, these types of solutions buy this flexibility, among other things, through a higher configuration effort, for example, in the form of policies that define appropriate data uses and privacy issues. In contrast, Figure 13.4b again states that most privacy-friendly applications are created for specific tasks, such as the evaluation of crowdsensed data focusing on participant’s locations [61, 92, 242, 264, 314, 405, 422]. However, solutions can also be found that provide only a framework for privacy-friendly data collection [87, 183, 238].

Table 13.2 Matrix illustrating the combinations of the data retention and the applied privacy-preserving method across the document corpus.

		Raw Data Retention		
		Local	Remote	Policy
Building Block	Cryptography	62*, 405†, 116†, 314†	422†, 263*, 87†, 46†, 116†, 61†	62*
	Monitoring	308*, 145*, 424*, 165*	145*, 81*, 266*	308*, 145*, 424*, 81*, 266*
	Obfuscation	239*, 183†, 308*, 405†, 360*, 330*, 164*, 306*, 92†, 238†, 240†, 116†, 314†, 242†, 410*, 31*, 218*	239*, 422†, 164*, 306*, 116†, 61†, 81*, 218*	308*, 306*, 410*, 81*, 218*

continued on next page

Suppression	308* 360* 147* 306* 145* 424* 410* 31* 218*	263* 306* 145* 81* 218*	308* 306* 145* 424* 410* 81* 218*
--------------------	---	----------------------------	---

* Standalone PET † Privacy by Design Architecture

In total, we identified four different building blocks to reduce the privacy risk associated with sensor data (c.f. Table 13.2). First, *cryptographic* operations can be applied to the data to limit the addressees. This building block includes symmetric [62, 116, 263, 405] and asymmetric operations [61, 87, 116, 263, 314, 405]. Homomorphic encryption [46, 264] or hash functions [62, 87, 263] are also used in the work. It is noticeable that the majority of research based on cryptography comes from the field of Privacy by Design. This is not surprising since architectures developed from scratch allow the inclusion of the security factor right from the start. It is significantly harder to use encryption in an existing multiparty application when the other party needs the data in raw form (as is the case with UBI, for example). Next, *monitoring* techniques are common in PETs and provide audit, log or notification functionality (these are also found in TETs). For example, a user is notified when an application accesses specific sensors through a notification led [266]. Subsequently, this access can be approved or denied, for example. This leads to the next class, *suppression*, which can be seen as a companion to the monitoring class. In such systems, users usually can allow or deny access to specific sensor readings (i.e. filtering). As shown in the matrix, they can be supported by a policy that takes care of tasks according to predefined rules. As an action to control access, *obfuscation* methods are available in addition to completely prohibiting the processing of sensor data. Here to mention are suggestions like the perturbation and generalization of data and mocking (partial or full insertion of synthetic data).

Protective measures

Data retention describes the location of sensor data. Table 13.2 presents three different classes as columns. The decisive factor for the classification is the location of the raw data; if the data has been anonymized, the work is not shown in the respective column. Data can be stored and kept *local* on a user’s device in contrast to data being submitted to a *remote* instance and kept there. Work in the *policy* column has a special status since the location of the stored data can be controlled flexibly. If necessary, it cannot be ruled out that raw data leaves the local device if the purpose of use makes this necessary; hence, such works are included in the *remote* column (e.g. [424]). Within Luo et al. [238], raw data is stored locally, but obfuscated data is forwarded to a server, which is an example of good practice. Comparable [31, 360, 410] are policy-driven

Data retention

systems in which the level of data leakage to a remote party is handled in a controlled manner. For instance, either perturbed sensor data is sent, although synthetic data may be used as a replacement if a constraint from a policy may not hold even for the anonymized data. This is preferred, as raw data is always kept locally within a user's trusted domain. However, there are some examples where data confidentiality and, eventually, privacy cannot be guaranteed. As shown in Table 13.2, approaches that are configurable using a policy sometimes allow data leakage to occur if the policy does not prevent data from being submitted [218, 240] They are briefly discussed in the following.

- ▶ Wu et al. [405] submits anonymized data which might not be sufficient w.r.t. side-channel attacks that we have seen before (e.g. our driver identification approach; Chapter 11 or trajectory reconstruction; Chapter 12).
- ▶ Next, Dua et al. [116] and Luo et al. [239] store raw data on a service proxy that, according to their design, may be a TTP. However, this is still considered a data leakage as raw information is leaving a user's device. This is a behavior that is found multiple times [164, 263].
- ▶ Then, the approach of Biswas and Vidyasankar [46] removes identifies from the data, which is also equivalent to a data leak as the raw data has left the device and is thus vulnerable to side-channel attacks.
- ▶ Miao et al. [264] and Shen et al. [340] store encrypted but non-anonymized data in the cloud that induces similar threats as in the previous example.

13.3 Conclusion

SLR outcomes This chapter generated an overview of state-of-the-art procedures to privacy-enhance the sensor data processing of user data. During the investigation, a considerable difference was found between the PETs and Privacy by Design architectures. This can be seen in the interactivity, where the latter mostly functions transparently, and also in the building blocks used for the protection concept. If cryptographic building blocks are dominant in Privacy by Design architectures, PETs usually aim to inform the user and change the quality of existing data or prevent access. Preventing access, however, contradicts Cavoukian [66] view that applications should still provide all functionalities even when privacy is considered. We have also seen that some proposals, in terms of their architecture, require a TTP, which is considered the central protection

authority. This may as well receive raw data, which should be critically viewed w.r.t. privacy (c.f. [126]). Better suited are solutions where raw data is kept locally, and only privacy-friendly (i.e. aggregated) data leaves a user's personal domain.

The findings are intended to serve as the basis for our PET's own proposal that specifically addresses the needs of UBI and, particularly, PHYD. The following proposal thus fits into those PETs that have been designed especially for one application area (e.g. location-based data) but still be agnostic to the business case as long as it is built on the processing of sensor data. Subsequent inclusion of protections excludes some identified building blocks for privacy. For example, encryption blocks cannot be used because the opposite party would not be able to process such data. The use of signatures as another form of cryptographic building block is possible since they extend some system properties but do not change existing ones. Proposals to generate synthetic data [360, 410] are not applicable within the framework of UBI since insurance companies would distort and thus reject the price of the driving style. Adaptability through a policy is desirable so that different stakeholders find themselves represented in the system accordingly. This should increase acceptance and flexibility. Furthermore, for the sake of comprehensibility, adoption, and effectiveness, a non-interactive PET should be provided that does not overburden the user (see bounded privacy; c.f. Acquisti [2] and Laufer and Wolfe [226]) and significantly shifts the responsibility for his privacy back to him. Instead, it should be supportive and should not cause any additional workload. The use of a data broker (i.e. TTP) to manage and control privacy without technical traceability should be avoided. If a TTP is necessary, it should not have the capability to process data accordingly without the user's consent.

Desired
properties of
a PET

kUBI: Aligning Privacy and Integrity in Sensor-Based Business Models

14

In the context of UBI, the driving behavior of people is used as the basis for pricing. Sensor data recorded during vehicle movement serves as the underlying foundation. This includes, for example, acceleration and braking characteristics or external factors such as driving time. We already learned about the corresponding business model and processes in Chapter 9 with a dedicated smartphone app that collects sensor data of trips that an insurer analyzes to derive a rating for each trip eventually used for pricing. Furthermore, in Chapters 11 and 12, it was shown that the sensor data used, which is often transmitted in the form of raw data, does not meet the privacy requirements of the individual. The data is often submitted in a non-transparent manner via the insurer's app (see Section 9.3) without control or intervention by the user.

Appropriate technical measures should be available to empower and educate users. The use of PETs and TETs lends itself to this. Due to the restrictive limitations and the necessary acceptance of users, but also of insurers, attention should be paid to the context of UBI (especially PHYD). Users are entitled to privacy, insurers to integrity. A similar issue was considered in the work of Part II, but as privacy-by-design there. In the previous Chapter 13, it was demonstrated that by including the privacy claim in the initial development, other technical options could be used.

Recap

This chapter introduces *kUBI*, a PET and TET, which meets the contradicting requirements of privacy and integrity. It facilitates anonymization of the user

kUBI

Acknowledgement

Parts of the research presented in this chapter are based on work published previously [318].

data in such a way that an evaluation in the sense of PHYD is still possible. At the same time, the proposal prevents user fraud (to gain a cost advantage) by requiring the integrity and knowledge of the raw data to be proven upon request by the insurer. Although we use the example of UBI throughout this chapter, *kUBI* is agnostic to the use case. Hence, it can be adopted and applied to different contexts of sensor data processing.

Contribution A holistic framework is presented that allows the preservation of privacy in the UBI context by applying the concept of k-anonymity to this domain. To the best of our knowledge, we are the first to adopt this concept in the field of UBI. Therefore, *kUBI* is short for k-anonymous Usage-Based Insurance. In fact, we

- ▶ present our comprehensive framework *kUBI* that takes into account the conflicting demands of the various stakeholders,
- ▶ discuss how to embed *kUBI* in existing UBI workflows,
- ▶ present a draft for implementing the approach in the Android architecture to enable privacy-preserving sensor data processing, and
- ▶ evaluate the framework thoroughly using the example of PHYD with real-world data and the attack from Chapter 11 to prove that *kUBI* is able to increase privacy and integrity.

Structure First, in Section 14.1 we present work that also focuses on increasing privacy. Subsequently, we present *kUBI* in Section 14.2 as an overview and address boundary conditions. A detailed discussion and description of all components follow in Section 14.3, before evaluating the suitability of *kUBI* based on attacks from Chapters 11 and 12 (see Section 14.4). We conclude this work in Section 14.5.

14.1 Related Work

Existing approaches PETs were focused on in the Structured Literature Review presented in Chapter 13. We found that most PETs were integrated into the data processing itself, for example, as a client-side component or performed alteration of the data locally. Such approaches are infeasible since there is no guarantee that the specific patterns evaluated within the PHYD model are intact. Furthermore, encrypting the data locally and sending aggregated results requires knowledge of remote data processing, i.e. how an insurance company evaluates events to calculate the premium eventually. Therefore, we propose a generalization-based approach

that adapts the idea of time series segmentation [204] that the insurer drives. In addition, it integrates the idea of a one-way function similar to hashing with its given properties to protect raw sensor data.

Also, around 20 % of the analyzed PETs require the application of a Trusted Third Party that receives the data to remove any identifying information before forwarding it to the service. As we have already discussed in Chapter 6, TTPs only shift the trust problem. Moreover, removing identifying information while keeping the raw data is infeasible in the given model since the insurance company needs to assign a rating to a vehicle (or, in general, a contract). Hence, pseudonymity is not a meaningful solution.

Moving trust

There is little work that concentrates on the topic of UBI, although these types of insurance models are gaining popularity. Troncoso et al. [373] has already provided a model for PAYD that is privacy-enhanced. PAYD is yet significantly different from PHYD, as it is often the location that is problematic [149]. Thus, the necessity for privacy-enabled PHYD models persists, given the prevalence of PHYD-enabled rates in the UBI business model.

Unsatisfactory solutions for UBI

Hence, we can only use selected building blocks or general ideas obtained by the SLR, but we have to accept that there is no suitable solution. Furthermore, most PETs rely on a user to make meaningful decisions about forwarding or keeping data, not taking into account aspects such as bounded rationality (c.f. Section 1.2.3). Policy-driven systems may be preferable to provide meaningful data quality. Policy-driven systems must support multiple stakeholders [56] to be applicable in the given context, and the policy must be verifiable for transparency reasons.

Lessons learned

Framework

14.2

The framework is designed to support the contradicting requirements of different stakeholder compositions; still, the proposal is device and application-agnostic to support various use cases and settings in the sensor-driven scenario. It is framed to be easily retrofittable to existing business models, making them privacy-friendly without altering the process itself. There is no focus on either one of the two parties, insurer and policyholder, as both should be equal participants in the business process.

14.2.1 Stakeholders

In the UBI scenario, three participants have been introduced: the insurer, the policyholder, and additional drivers, even if the last two can be combined. It is self-evident that all sides have divergent priorities regarding the business model and its everyday use.

Insurer The insurer is primarily concerned with accurate and correct data to draw appropriate conclusions. In particular, he assumes that the submitted sensor data is with integrity and collected using his infrastructure, i.e. his application. Further distinctions can be made between two aspects of integrity. First, an insurer is concerned with accurate data (*data integrity*) in order to determine the proper categorization of drivers for a trip. The data has to contain the whole trip, should not be prerecorded, nor should any preselection of data patterns be performed. Second, *system integrity* is critical to ensure that the workflow is consistent with the business model. However, system integrity is a hard-to-grasp concept since the data processing pipeline to eventually derive a trip rating for the submitted data is confidential. Hence, it might be unacceptable for an insurer to receive data via third-party applications, especially not in an altered shape.

Policyholder The policyholder wants to protect his *privacy* so that only the business model can be applied, but data misuse cannot occur (based on attacks such as presented in Chapter 10). PHYD's business concept collects data such as GPS coordinates or sensor data from the accelerometer or gyroscope while the user is driving to provide different insights into the user's driving style. One could argue that these data are crucial for privacy purposes. According to Pfitzmann et al. [297], privacy is conveyed, among other things, through users' ability to determine which data is shared and to what extent. This is not the case for the majority of PHYD models, as the data is obtained according to the insurer's specifications, sometimes being raw and unbound data (see Section 9.3). However, a user is eligible for a discount only if he sends data. In addition to privacy, users also care about the integrity of the sensor data submitted, as it is the basis for the premium calculation.

Additional drivers Additional drivers have similar interests as the policyholder. They require privacy from the insurer and, on the other hand, also within the group. Thus, no other driver (including the policyholder) should be able to make statements about an individual's driving behavior. This corresponds to the requirements and wishes established by Solove [350]. Each person should be responsible for his behavior without being discriminated against by others based on their driving behavior.

To establish trust in the system, specific relationships between stakeholders must be considered [377] which will be discussed later.

Strategies

14.2.2

Various strategies can reconcile the participants' different views and expectations within the framework of existing business models, ultimately increasing privacy. The challenge is to choose a strategy that does not overestimate the respective interests of one party to the detriment of the others. Furthermore, the solution should not unnecessarily complicate the process or the framework conditions, such as infrastructure. In concrete terms, the four strategies are conceivable, namely processor-based, user-based, trustee-based, and balanced. Now we explain every strategy.

Processor-based The data is collected according to the sole definition of a data processor (typically the insurer) by users using a dedicated device or a smartphone application. Data is then transmitted to the processor for centralized evaluation. The user cannot control, nor is there any technical proof that the data is only used for a specific use case, resulting in a less trustworthy model. Additionally, it is not mandatory to communicate which data is transmitted or how the evaluation is performed. Therefore, this model is a classic example of *covert trust*, where an insurer wants to protect its interests from the legitimate user of the device. Trust is high from a processor's perspective as he controls the whole data pipeline with his infrastructure. This is the classic model found in the context of UBI.

User-based All information and data are processed and analyzed within the user's local device. This strategy, according to Pfitzmann et al. [297], is an example of *personal agent trust*. Since a user's own domain is the most trusted in the scenario, this way is preferable in terms of privacy so that only aggregated and earmarked information leaves this domain. It is preferable for transparency but optional that the user explicitly approves the information submitted for transparency. Integrity is more difficult to verify when user-processed information is received on the processor side.

Trustee-based Although trust is transferred from the processor to a trustee (i.e. a TTP), this represents a marginal improvement for the user over the processor-based method [126]. Confidence in a model is not raised, but another participant is introduced who gains access to sensitive user data. The business model and respective pipelines are not altered.

Balanced Data processing is carried out in a hybrid way, involving both the user and the processor so that both interests are equally taken into account. The processor establishes the required data quality by using a policy that a user must respect to participate in the process. The customer anonymizes the data following the policy and submits only aggregated findings. Additionally, a user can ensure that his or her own private goals are met while maintaining anonymity, as he can abort the submission process. Thus, while the process can regulate integrity, the user is entirely responsible for the privacy produced in its trusted domain.

Within the outlook of this dissertation in Section 15.3, we introduce the split trust continuum [363]. This construct explicitly targets the situation where trust between stakeholders is missing or limited. Such a situation may also be present in the current case; hence, the split trust continuum may be a viable solution to tackle this fundamental problem of who should verify the the system's general properties such as a policy.

14.2.3 *kUBI*

We can now integrate the previous considerations into our holistic privacy-enhancing framework *kUBI*. As mentioned above, PHYD is used as a running example in this chapter. However, *kUBI* was created so that it can also be transferred to other scenarios and used there. The condition is that data collection and analysis can be separated accordingly, as is the case here (smartphone for collection, central infrastructure for analyzing).

Preliminary
considerations

Based on stakeholder analysis, we define that a holistic solution to enable privacy-preserving sensor data processing within business models requires a combination of PET and TET. A PET is required to enable privacy in the first place and empower individuals to anonymize data and to selectivity decide which data leaves their private domain. However, a user may not understand what data is submitted and how it is evaluated without knowledge of the data and education on the processing. Thus, the aspects of a TET are included in *kUBI*, and a user will be able to see the events that the insurer will process. We chose a balanced strategy since it incorporates all stakeholders' interests equally. Arguably, it is meaningful from the insurer's perspective to make such a compromise, as the willingness to share data increases and privacy concerns decrease with active user participation in the process [88]. However, the information is published through a data processor-defined environment without requiring an external TTP, as proposed by others [194].

In order to preserve the interests of the different stakeholders, the process is divided into exclusive domains according to Pfitzmann et al. [297]. The main idea is to balance the risk of, for example, fraud and the amount of trust that must be placed on other parties [160]. The proposal has two primary areas representing the two stakeholders: the user and the processor. One is the local trusted zone located on the user's side, in the form of his personal mobile device (*User Domain*). We assume that this is his smartphone. The other zone represents the infrastructure of the data processor, i.e. the insurance company (*Business Domain*). The respective parties do not trust the counterpart's processes in the corresponding other zones. However, two exceptions reintroduce overlapping trust (*multiagent trust* [297]) into the system. First, the *Hostile Domain* (which may be a Trusted Execution Environment (TEE)) allows trusted and integrity-preserving execution of foreign code components in an isolated hostile environment in a predefined way so that internal operations are hidden from the user, eventually being unable to be tampered with. An TEE, in general, has well-defined input and output variables and is limited to no communication capabilities. The second zone with higher-level properties is the *OS Domain* which is a protected area of the Operating System that provides functions to application developers via the defined APIs. Looking at the Android sensor stack, the OS domain functionality necessary for *kUBI* can be located in the Hardware Abstraction Layer (HAL) [23] with the HAL being an interface between the hardware sensor and the Android framework (see Sidebar H). Hence, the OS domain is considered the lowest level of the *kUBI* proposal and enables application developers' initial access to the sensor data.

Isolation of knowledge

Android Stack

Sidebar H

We present the sensor stack using Android as shown in Figure 14.1 based on Android Open Source Project [23]. Developed applications communicate with Android OS using the Software Development Kit, which provides a high-level overview of the sensors. The *Framework* layer establishes a connection between numerous apps and the HAL. It introduces multiplexing, which enables concurrent access to a sensor by various *applications*. Within this layer, virtual sensors are also constructed.

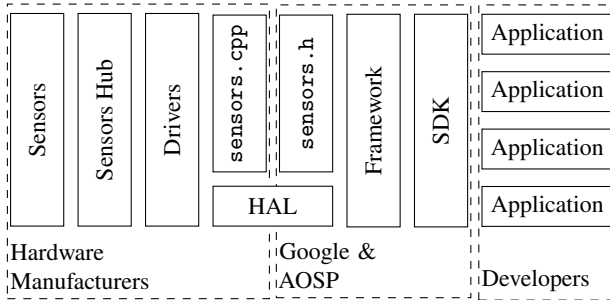


Figure 14.1 Layers of the Android Sensor Stack. (based on Android Open Source Project [23])

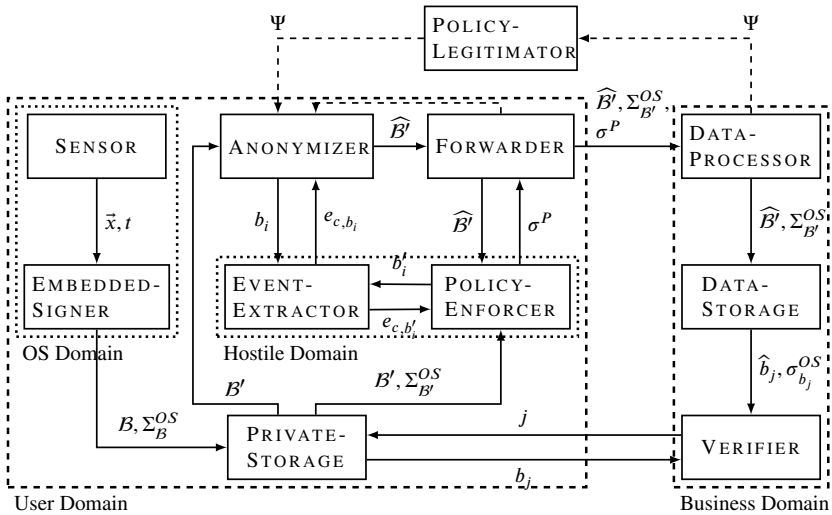
The HAL is the interface between Android and the implementation of a particular sensor by a hardware vendor. It adheres to a well-defined user interface (`sensors.h`). Lower levels are wholly owned by hardware makers and may be closed source. Android is unable to modify the behavior of any sensor unless it is described in `sensors.h`.

The stack is arranged bottom-up for security concerns, which means that higher levels cannot communicate data to lower instances (Figure 14.1 is flipped due to space constraints). Multiple applications that access data from the same sensor operate independently. It is straightforward to register sensor readings.

To begin, an Android-wide `SensorManager` provides access to the `SensorService`, which can be used to access a variety of sensors. Then, to handle sensor updates, a `SensorEventListener` may be utilized. The code snippet shown in Listing 4.1 queries the accelerometer and keeps track of newly obtained `SensorEvents`.

K-anonymity

It is challenging to provide privacy within the domain of PHYD as information on the intensiveness of events or events at all should not be missed or altered, as this will prevent the assessment of a trip from calculating the premium. However, as we have learned in Section 9.4, it is sufficient for a data processor to assess events \mathcal{E}_c to specific categories $\gamma_{e,1}, \dots, \gamma_{e,n}$, for example, aggressive, neutral, and passive, using an unknown function $C : \mathcal{E}_c \rightarrow \Gamma_{e_c}$ with Γ_{e_c} being all event categories (c.f. Chapter 11). *kUBI* explicitly protects a user's privacy by reducing the level of detail provided to the sufficient degree that a processor must receive. Our proposal eliminates the slight differences between drivers within



Framework for gathering and processing sensor data from users in a corporate domain in a privacy-enhanced way for existing business models. The framework posts different domains that are used to either generate trust, enable privacy or provide integrity.

Figure 14.2

the same event category by providing *k-anonymity* within such semantic groups using the idea of time series segmentation [204]. Still, the event extraction and expressiveness of an event will be present, as *kUBI* will be orchestrated using a processor’s provided policy Ψ and replaces a driver’s events with corresponding reference events provided (i.e. segmented time series). Reference events, denoted as $\tilde{\Gamma}_{e_c}$, are provided by the processor in the form of Ψ . They may be derived from the barycenter calculation of historically collected events (see Sidebar G) or any other “magic” responsible to the processor. This prevents unnecessary identification by the processor, for example, with the presented identification attack (see Chapter 11). *kUBI* collects a complete sequence of events $\mathcal{E}_{c, \mathcal{M}}$ from a trajectory \mathcal{M} by applying CEP. Eventually, Ψ contains the necessary parameters for processing the time series data, for instance, window length or window overlap. All events $\mathcal{E}_{c, \mathcal{M}}$ are then surrogated in a way with appropriate drop-in replacements from $\tilde{\Gamma}_{e_c}$ so that no subsequent conclusions can be drawn about a driver. It is not practical to do the entire trip assessment in the user domain and then transmit the information to an insurer since supplementary or historical knowledge is required and computational resources are limited.

Architectural
overview

The proposed framework *kUBI* is composed of different components separated into four different domains, as illustrated in Figure 14.2. The user domain (i.e. mobile phone) includes the hostile domain and the OS domain with well-defined flows among each other and to the external business domain (i.e. insurer).

High-level
process flow

A hardware sensor `SENSOR`, such as the gyroscope, continuously generates sensor readings $\vec{x}_1, \vec{x}_2, \dots$ at specific times t_1, t_2, \dots . They are sent directly to `EMBEDDEDSIGNER`, which eventually wraps and signs multiple sensor readings discretely in a data block b without leaving the OS domain. Hence, data blocks cannot be changed afterward due to a cryptographically-secure signature σ using OS-provided key material. The data block-signature pairs are persisted in a `PRIVATESTORAGE`, which is a starting point for further processing. Consider a data block representing a trip B' that is a subset of all data blocks B . B' is the trip about to be submitted to the processor and, therefore, forwarded to two modules, the `ANONYMIZER`, and `POLICYENFORCER`. The `ANONYMIZER` is responsible for replacing or altering events $\mathcal{E}_c \in B'$ w.r.t. the provided policy Ψ to ultimately raise privacy for users by transforming the sensitive trip B' into an anonymised version \widehat{B}' . Since *kUBI* is not aware how to extract \mathcal{E}_c due to confidential reasons illustrated previously¹, the user requires the utility of the processor to provide an event extractor, namely the `EVENTEXTRACTOR`. It is embedded in the hostile environment and, according to his internal and secret definitions, is initialized by the insurer. However, the replacement or alterations of events can be performed without additional knowledge. The anonymized trip \widehat{B}' is sent to the `FORWARDER` that uses the `POLICYENFORCER` (also in the Hostile Domain) to verify acceptable and feasible anonymization by the `ANONYMIZER`. The `POLICYENFORCER` confirms an integer process by issuing a signature σ that is submitted by the `FORWARDER` along with \widehat{B}' to the `DATAPROCESSOR` with the consent of the user (thus providing autonomy to the user [297]). The data processor verifies the respective signature of an incoming trip to legitimate it coming from an integer system. It is processed in a black box and stored in the `DATASTORAGE`. A feedback channel (`VERIFIER`) allows the processor in a limited process to request unmodified data blocks from a user, if necessary, to verify the integrity of the process and the transmitted data.

¹An insurer does not need and probably will not publish his underlying event extraction method E_c ; c.f. Chapter 9

Policies

14.2.4

We chose a balanced strategy for *kUBI* that combines privacy enhancement within the trusted domain of a user and integrity validation by relying on the definitions and infrastructure of an insurer.

The core of the balanced strategy is the policy that defines and constrains the level of anonymization applied to enforce privacy. Therefore, it is important that the policy is *feasible* in terms of still allowing *integer* evaluation of PHYD, being *sound* so that data alteration methods are realistic, and being *acceptable* to provide a sufficient level of privacy. The policy is fundamental for the autonomy of users to gain back empowerment from the insurance company [407].

Core
functionality

The verification of the policy and the entire process requires a thorough review. Three levels of verification are conceivable: verification by the provider itself, an intermediary, or the user himself. These are discussed below.

Individual Users can perform the policy review themselves. They do not depend on any other entity for this and enjoy *full authority* in their decision-making. The protection techniques applied in the system are defined and verified by the user so that the user himself can adjust the level of privacy. Users are free to decide whether the policy meets their privacy needs and make an informed decision, provided they understand the situation accordingly. This represents the ideal solution but is challenging to implement due to the complexity of the process and the required understanding [226].

Intermediary An independent third party can review the policy on behalf of the users. In contrast to the individual review process, part of the authority of the users is relinquished in favor of increasing convenience and reducing complexity (*limited authority*). The transparent review process takes over the review tasks by experts and professionals who understand the subject matter and can thus perform an in-depth analysis. They represent users' interests (such as privacy and anonymity) and can, for example, provide recommendations and advice to users. Due to the proxy principle, intermediaries represent many users and can seek vigorous discussion with processors should privacy disagreements arise or the policy allows inappropriate data processing. Examples of intermediaries are data protection commissioners of federal states or consumer centers (Verbraucherzentrale)².

²https://www.bmfv.de/DE/Verbraucherportal/Verbraucherinformation/VZBV_VZ/vzbvundVZ_node.html

Processor The processor publishes the policy according to its conditions and philosophy. It can be assumed that the policy is drafted according to the legal framework (e.g. GDPR), but its own interests are given priority, such as thorough data knowledge. Although the process has complete knowledge of the data processing procedure, the correctness of the policy is confirmed solely by the creator, which implies *no authority* towards the provider by the customer.

Each level of verification has the corresponding advantages, be it complete self-management of the users (**Individual**), the streamlined process of verification by experts (**Intermediary**), or complete knowledge of the processing steps of the sensor data (**Processor**). *kUBI* supports all three levels in its design, as the policy is an external artifact provided to the ANONYMIZER. The syntactic correctness of the policy is essential primarily for the functionality of *kUBI*. Verifying the semantics is possible according to the three levels presented and is optional. The processor level can be considered the default if the verification is omitted.

14.3 Components

We now explain every component in-depth³. The explanation will follow the process flow of the framework. We start with the OS domain, then explain the core *kUBI* building blocks with the user domain, including the hostile domain, and finish with the black-box-like business domain.

kUBI is a policy-driven system that can be configured in a compressible way by a data processor via Ψ . This policy is subject to review as it is fundamental to the acceptance and feasibility of a privacy-enhanced business model. A POLICY-LEGITIMATOR symbolizes the review process. We have already discussed different characteristics for this component in Section 14.2.4; hence, we neglect further remarks.

14.3.1 OS Domain

The OS is only under the control of the device manufacturer or the OS vendor. We assume that no alterations can be made to this environment except by the vendor itself.

³Although this section is based on Roth et al. [318] with several extensions; the notation has been adapted to fit the notation in this work, which is used throughout this work. Consequently, there might be differences compared to Roth et al. [318].

Sensor

Based on the information provided in Chapter 2, we assume that a PHYD application requests data from the sensors of interest, such as the accelerometer or gyroscope using the official APIs. A generic `SENSOR` continuously outputs a stream of measurements $\vec{x}_1, \vec{x}_2, \dots$ with the shape varying from sensor to sensor. According to our definition of time series data, this stream of *Sensor Events* is chronologically ordered by a respective timestamp t for each reading. For example, the a `SensorEvent` of the accelerometer is a three-dimensional vector along with a timestamp: $\vec{x} = (\vec{x}_x, \vec{x}_y, \vec{x}_z, \vec{x}_t)^T \in \mathbb{R}^4$. Notably, the data shape is not altered by *kUBI* to enable backward compatibility with existing applications. Furthermore, existing APIs will be used for interaction by privacy-enhanced sensor applications.

EmbeddedSigner

As we use the same interfaces as offered by Android for processing sensor data, an application can register at the `EMBEDDEDSIGNER` (or in terms of Android: Signed Sensor Event API) for corresponding updates of *Signed Sensor Events* extended with appropriate integrity-protecting fields in contrast to *Sensor Events* (see Chapter 2). The extension of the Sensor Event API to generate should be provided by the OS vendor itself as a drop-in replacement for broad and easy adoption. The Android Sensor Stack architecture enables the separation of data acquisition utilizing the physical sensor and data processing in the application; hence, compatibility of the proposal is ensured.

Since cryptographically signing is an expensive operation, it is minimized by concatenating multiple *Sensor Events* based on a time frame ω , for instance, 1 s. It is important that this does not reduce the frequency of sensor readings to 1 Hz as a `SignedSensorEvent` serves as a container for holding a sequence of `SensorEvents`, allowing access to all successive sensor readings. They are defined as a data block b_m with n elements:

Data blocks

$$b_m = \left\langle [\vec{x}_1, \dots, \vec{x}_n], \sigma_{b_m}^{OS}, GUID \mid x_{l,t} \in [t_s, t_s + \omega] \text{ for } l = 1, 2, \dots, i \right\rangle$$

t_s denotes the beginning of b . Data blocks are discrete and do not overlap, i.e. the $m + 1$ -th data block has \vec{x}_{n+1} as its first sensor reading. Attached is a signature $\sigma_{b_m}^{OS} = (\text{H}_{OS}(\vec{x}_1 \parallel \dots \parallel \vec{x}_i))^{d_{OS}}$ that provides integrity and authenticity. H_{OS} stands for a secure and publicly known hash function that is used to hash the concatenation of all subsequent `SensorEvents` included in the timestamp, while d_{OS} is the `EMBEDDEDSIGNER`'s private key used to sign the signature.

Furthermore, we add a Globally Unique Identifier (GUID) to each data block which ensures that a data block can be identified uniquely and without ambiguity in a set of data blocks, although this id is not related to the payload.

Hierarchical
trust

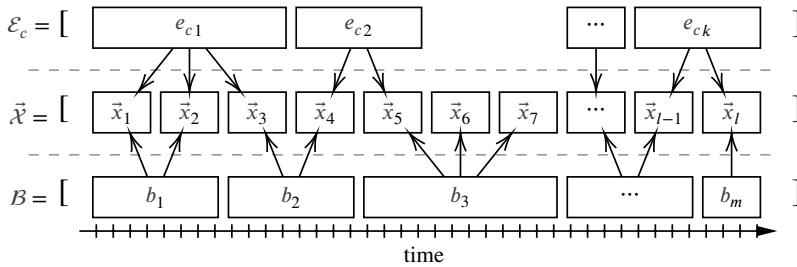
The EMBEDDED SIGNER is located in the isolated OS domain, eventually being protected from user interference. Android already provides secure key management via the Keystore API, which is embedded in a TEE isolated from the userland application and is used to produce and verify signatures. Therefore, the OS vendor can serve as a root trust anchor trusted by all participants of the process (*multiagent trust*), essential for building a PKI. It can deposit a device-dependent key, for example, d_{OS} securely in the device's Keystore for operations over the Keystore API. The trustworthiness of the key is appropriately confirmed by it. Ultimately, data signed with the device key is to be trusted because the corresponding root CA, i.e. Google, is assumed to be trustworthy (i.e. being a hierarchical trust system). The public certificate for the device may be widely published, for example, to enable a data processor to validate signatures.

14.3.2 User Domain

The user domain runs userland applications on behalf of the user without any guarantees regarding security and integrity towards a third party. A user may tamper with applications and data under his control and interact with the system in a manner primarily intended by him. *kUBI* uses components in this domain to enforce a user's interests in the business model.

PrivateStorage

The persistent storage in the user domain called PRIVATE STORAGE holds all data blocks \mathcal{B} recorded by a device. Apart from \mathcal{B} , the set Σ^{OS} of all signatures $\sigma_{b_1, \dots, b_{|\mathcal{B}|}^{OS}}$ is stored alongside to be able to prove authenticity and integrity of all blocks. The storage also organizes trips that are defined similarly to Chapter 11: A trip is a semantically coherent sequence of measurements without interruption, such as those occurring during a journey from A to B. Hence a trip \mathcal{B}' is a semantically sliced subset of \mathcal{B} , defined as the concatenation of multiple data blocks b_i, \dots, b_j with the respective signature set $\Sigma_{\mathcal{B}'}^{OS} = [\sigma_{b_i} \parallel \dots \parallel \sigma_{b_j}]$. \mathcal{B}' s are processed and classified *ex-post* by an insurer, i.e. after the trip has been completed. This component is also relevant for the legitimation of the transmitted data. With its help, a user can prove that he knows the corresponding genuine data blocks for the anonymized data blocks.



\vec{x} are organized in B and contain patterns representing \mathcal{E}_c . All elements are interchangeable and are time series based. Figure 14.3

EventExtractor

According to the PHYD model, an insurer classifies a trip based on certain events that occur during the journey. The assumption is that these allow a conclusion about the driving behavior. This hypothesis was confirmed by the attack featured in Chapter 11. Thus, the events are the baseline for the anonymization, implying that the events are appropriately known, or it is at least known how to derive them based on a trip. Consequently using a set of events \mathcal{E}'_c in a trip B' , an anonymized version of the trip \widehat{B}' can be generated according to communicated policy Ψ . However, the process of extracting events is unknown. We assume that only an insurer is aware of a confidential method E_c . *kUBI* takes this into account by embedding E_c in the form of the `EVENTEXTRACTOR` in the system's hostile domain. The `EVENTEXTRACTOR` works as-a-service that adheres to black-box principles in order to protect the insurer's business case (*undercover-agent trust*), i.e. there is no need to explain or publish. This ensures confidentiality, a critical component identified within the stakeholder analysis. As a result, the `EVENTEXTRACTOR` uses a proprietary procedure to extract events from the data block stream. However, the user retains control over input and output, which contributes to increased trust. He is solely interested in the projection of data blocks to events to get all events \mathcal{E}'_c in a trip B' : $b_i, \dots, b_j \rightarrow e_{cm}, \dots, e_{cn}$ with $\forall b \in B'$. The `EVENTEXTRACTOR` yields a list of events based on all data blocks with start and end times.

Hidden event extraction

Events are ultimately linked to data blocks that, in turn, are composed of sensor readings via the `EMBEDDED SIGNER`'s concatenation algorithm. Since sensor readings are time series data streams, all successive elements built on top of them can be addressed by the same time information. The link between the three elements is illustrated in Figure 14.3. Sensor readings do not necessarily

Transformation between elements

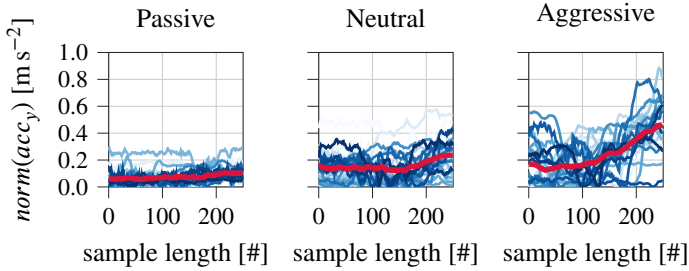


Figure 14.4 Reference events $\check{\mathcal{E}}_{c,i}$ with $i = 1, \dots, v = 3$ are representative for a category. We assume that reference events are the weighted average of already collected events $\mathcal{E}_{c,1,2,\dots}$ for a category. Shown are events of type *braking* for a single \bar{x} (ρ is set to 250) which are deployed by an insurer and then used locally at the user.

form an event, e.g. if a driver is only going straight without accelerating or braking. Hence the projection of events and data blocks is non-surjective and also non-injective as a single event may be composed of multiple data blocks, but not every data block is included in an event. However, the link between sensor readings and data blocks is a non-injective surjective function, as every sensor reading is contained in a data block, although one data block contains multiple sensor readings. Let us define two utility functions $O_{\bar{x}}(e_c)$ and $O_b(e_c)$ to count the number of sensor values $\bar{x}_1, \bar{x}_2, \dots$ or the number of data blocks b_1, b_2, \dots (with equal length in terms of time), respectively, that compose an event e_c . It is likely that $\exists e_{ca}, e_{cb} \in \mathcal{E}_c : O_{\bar{x}}(e_{ca}) \neq O_{\bar{x}}(e_{cb})$ holds. Elements can be converted from one type to another, as illustrated.

Anonymizer

We initially motivated the privacy issue of PHYD by showing that an insurer can gain additional knowledge based on the extracted events \mathcal{E}'_c . A proof is presented in the form of the identification attack in Chapter 11, particularly stressing the importance of anonymity. However, precisely these entities are used to classify a trip. Thus, the integrity of events is of crucial importance. Furthermore, sending raw data to foreign domains is prohibitive for the reasons mentioned earlier, and the ANONYMIZER is responsible for balancing the mentioned interests. *kUBI* balances both stakeholder interests by replacing identified \mathcal{E}'_c in a privacy-friendly, but comprehensible way with so-called *reference events* using a one-way function with similar properties as a hash function $B' \rightarrow \widehat{B}'$.

A reference event is a representative of an event type of a class that can be defined using the policy Ψ that is first introduced in the context of PHYD. The reference events are to be constructed by the data processor, the i.e. insurer, based on his understanding of how a class of an event may look like. As they are static and independent from the current circumstances, such as the activity of other drivers, the ANONYMIZER, therefore, does not need an interactive interchange with the data processor to, for instance, gain any global noise [183]. They are syntactically identical to events already introduced in this work but are a generalized view of the problem; hence, a generalization-based approach is pursued to enhance privacy. We may assume that reference events are constructed based on real events by overlaying them, which is a typical attempt to generalize a problem in speech recognition, e.g. using barycenter computation (see Sidebar G). However, *kUBI* is independent of the construction of the reference events $\check{\mathcal{E}}_c$ since these are similar-shaped time series data excerpts as introduced previously (e.g. \vec{x} , or in particular \mathcal{M} in terms of PHYD). A reference event may have the same dimension as an event extracted from the measurements. In particular, $\check{\mathcal{E}}_c$ is a $\mu \times \nu$ matrix where μ is the number of classes that an insurer uses to categorize a trip and ν is the number of reference events that can be selected to replace an event class. We recall that in the given scenario, $\mu = 3$ classes exist and that in the simplest case only one event per class is selectable ($\nu = 1$). Figure 14.4 illustrates an example of how reference events can be constructed from previous, globally recorded events of a category in the processor's domain using the weighted average. Another approach might be to define an arbitrary polynomial function that yields representative sensor values.

Reference
events $\check{\mathcal{E}}_c$

The component in the user domain processes every $e_c \in \mathcal{E}'_c$ to select a suitable reference event $e_c \in \check{\mathcal{E}}_c$ using a distance-based approach, precisely DTW as the elements are shaped similarly (c.f. Sidebar F). The use of a distance-based metric eliminates the need for further domain knowledge within the ANONYMIZER, ultimately reducing the problem of selecting the correct $e_c \in \check{\mathcal{E}}_c$ to a minimization problem. In particular, the component does not need any information on μ or how to calculate them. The events of a trip are first resampled using R to the same length as is the case for $\check{\mathcal{E}}_c$ (i.e. ρ). After selecting the best-fitting reference event $\check{\mathcal{E}}_{cij}$, the inversion of R^{-1} must be found, so that $\check{\mathcal{E}}_{cij}$ can be interpolated in the sensor data stream by creating adequate data blocks and respective sensor values. The result of the ANONYMIZER is a set $\widehat{\mathcal{B}}'$ which contains the trip, but all events have been replaced (and thus anonymized to provide *k*-anonymity) with corresponding reference events. However, since the event (and its corresponding data blocks and sensor values, respectively) has to be adjusted to fit into the data stream, we call it \hat{b} . Note that the characteristics of

Event
exchange
process

a trip, its event class distribution, and order have not been altered at all, allowing the data processor to draw the same conclusions.

Other Ψ
approaches

Although we introduced reference events as a new generalization-based approach to tackle the complex problem of anonymizing time series sensor data in a somewhat constrained environment, other descriptions within the policy Ψ are conceivable. For example, known approaches to privacy-enhanced sensor data are suppression, randomness, alteration, or insertion. In Roth et al. [322], we analyze the respective methods, especially for the case of PHYD, but in the end none of them produces a sufficient level of privacy considering the required constraints. For instance, suppression is not feasible since the time series data within our identification attack (see Chapter 11) is resampled, eventually interpolating these missing values. Alteration or insertion of additional sensor readings tends to be unacceptable, as it may change the class of an event. This highlights the potential of the attack based on distance-based sensor data processing using error-robust methods. However, for other attacks without strict constraints, such as the reconstruction of the presented trajectory (see Chapter 12), a more straightforward policy Ψ may be sufficient.

Forwarder

Trust in the approach is i.a. based on the separation of knowledge and power. Consequently, the data is not automatically forwarded by an insurer application, but it leaves the user domain only through a channel defined and controlled by the user (*personal-agent trust* [297]). If a user decides to submit data to a data processor, he will need a valid signature $\sigma^B(\sigma_{\widehat{B}'}^{OS}, \widehat{B}')$ to demonstrate that the user followed the agreed anonymization process between the data processor and him according to Ψ . Therefore, a user forwards the anonymized trip \widehat{B}' to the POLICYENFORCER to have the correctness of the ANONYMIZER process confirmed.

PolicyEnforcer

The data processor cannot only define in a policy Ψ which reference events are available but also determine to what extent the sensor values may deviate between the original and the resulting trip⁴. The trip's data quality is ensured

⁴The ANONYMIZER uses a distance-based approach for the selection of reference events. In exceptional cases where the actual event is between two classes or is close to the boundaries, it may be replaced by a non-class reference event. This process cannot be recognized by the ANONYMIZER, because there is no insight into the business model. Therefore, it makes sense to consider this by allowing for a certain number of false positives.

by defining a policy regardless of the anonymization of the sensor data. The POLICYENFORCER plays a distinctive role in the system, as it preserves the integrity of the data and accepts the anonymization approach on both sides. First, it verifies that the anonymized trip \widehat{B}^l was created on behalf of the real trip data B^l . Recall that each $\vec{x} \in B^l$ has a timestamp that cannot be altered due to the signature $\Sigma_{B^l}^{OS}$. Therefore, this combination is used to prove the integrity of a data processor and is shown to him via a signature from its POLICYENFORCER. Let the POLICYENFORCER component provide functionality for crafting a valid signature if Ψ holds: $\sigma^P = \left(\Sigma_{B^l}^{OS}, H_P \left(\widehat{B}^l \right) \right)^{d_P}$ where H_P is a secure hash function that is authenticated using a signature created with the data processor's private key d_P securely stored in the POLICYENFORCER. The signature is required to perform further integrity checks within the framework essential for trust on the data processor side, as seen in VERIFIER. POLICYENFORCER is implemented in the hostile domain within the user domain for trust reasons. A user has no control over or is able to analyze the blackbox's activity. However, he does have control over the input and output settings, which is an important criterion for user acceptance. The box cannot communicate with the data processor through a side-channel due to the properties of the hostile environment (i.e. TEE). The output is auditable and cannot include concealed data.

Since the POLICYENFORCER is flexible in terms of the applied policy, we give an example of possible verification criteria hidden in Ψ . An anonymization may be correct if

Example

- ▶ the number of events in \widehat{B}^l (loosely) equals B^l ,
- ▶ input B^l from the PRIVATESTORAGE can be used to verify if the distribution of event types in the anonymized trip \widehat{B}^l equals B^l , and
- ▶ each $b \in B^l$ has a valid signature σ_b^{OS} , i.e. no recorded trip is used to deceive the PHYD system (replay attack); data blocks contain \vec{x} s which in turn have a timestamp \vec{x}_t signed in σ_b^{OS} .

For privacy reasons and to clearly separate the domains, the POLICYENFORCER does not have access to the ANONYMIZER, hence is unable to anonymize a given event in B^l to prevent any kind of oracle. Within this framework, the design of the policy is therefore deliberately limited.

14.3.3 Business Domain

The business domain is the counterpart of the user domain and is solely under the control of the data processor. It has to be assumed that all information leaving the user domain via a defined channel such as the FORWARDER is disclosed. This domain is responsible for processing the data, evaluating the trips, and pricing the premium. As mentioned previously, evaluation methods, including additional knowledge, are hidden from outsiders. Therefore, we consider this to be a black-box-based process.

DataProcessor and DataStorage

Upon receiving submitted and anonymized data blocks \widehat{B}' along with a corresponding signature $\sigma^P \left(\Sigma_{B'}^{OS}, H_P \left(\widehat{B}' \right) \right)$, the data processor first checks the validity and correctness of the signature σ^P . Recall that σ^P has been crafted according to the definition of the data processor within the POLICYENFORCER placed in the hostile domain.

Acceptance
test

A valid signature, therefore, confirms two properties. First, it allows the data processor to verify that the data originates from an instance that is under its control and thus legitimate. This is enabled by the fact that the secret key d_P used for signing is tightly embedded in the POLICYENFORCER and is protected from access by the TEE accordingly. Furthermore, it can verify that the transmitted data blocks \widehat{B}' have not been subsequently manipulated and that they match the trip to be examined. To accomplish this, the DATAPROCESSOR can compare the self-signed hash $H_P \left(\widehat{B}' \right)$ with the received data blocks. Thus, a user is forced to submit the data blocks that the POLICYENFORCER has checked. Otherwise, a fraudulent submission may be detected. Thus, replay attacks as a representative example are successfully mitigated. Subsequently, both \widehat{B}' and σ^P are persistently stored and evaluated according to the specific methods of the data processor.

Verifier

To further ensure the integrity of the process and to handle the case of suspicion of fraud by a user, the data processor has an additional option at his disposal. Recall that the ANONYMIZER performed a one-way anonymization operation $B' \rightarrow \widehat{B}'$ that is hard to reverse, as this introduces a specific amount of generalization into the data. To exclude fraud, the VERIFIER can require a

knowledge proof from the user, in which the user has to prove that he is the originator of the data.

Therefore, the VERIFIER selects an agreed quantity⁵ of random elements from the received trip \widehat{B}' (denoted as proof set $B_P \subset \widehat{B}'$) and requests raw data blocks from the prover, i.e. the user. Consequently, a trip is only considered valid if $\forall \widehat{b}_i \in B_P : \exists b_i \left[\text{PS} \leftarrow b_i \wedge b_{i,\text{GUID}} = \widehat{b}_{i,\text{GUID}} \right]$ holds, i.e. a user can submit the raw data block for any given anonymized data block as he can find it in his PRIVATESTORAGE (denoted as PS) and GUIDs match. The GUID is immutable and does not change independently of the anonymization process.

Knowledge of blocks

Furthermore, the VERIFIER must confirm that the provided event was not altered in terms of sensor readings or timestamps, as the GUID is not linked to the payload of a data block. $\Sigma_{B'}^{OS}$ holds all signatures $\sigma_{b_1, \dots, |B'|}^{OS}$ of the data blocks as processed by the EMBEDDEDSIGNER. Therefore, the VERIFIER can calculate the hash using H_{OS} of the raw data blocks submitted that he requested with B_P . He can then verify whether the signature derived from such a block is part of $\Sigma_{B'}^{OS}$. The proof is completed once the user can submit all requested data blocks and if each is part of the trip, which is verifiable thanks to the signatures.

Integrity of blocks

If any of the operations performed by the VERIFIER fail, the user is not actively supporting or able to deliver the requested blocks, ultimately the proof will fail. However, the user can also cancel the proof at any time if he finds discrepancies (e.g. the data processor requests more blocks than agreed upon), which ultimately supports his authority to participate in the process.

Proof failure

Discussion on Design Principles

14.3.4

kUBI is designed to balance integrity and privacy. These aspects are integrated at several points in the framework. We now discuss elements that support integrity and trust, as they are the dominant stakeholder interests. Both objectives are essential for sensor-based business models.

Integrity

The pattern design implements three different proofs needed for a privacy-balanced business model. A user is also interested in integrity because he only

⁵The quantity is a critical component that requires careful discussion as it is a security parameter to balance integrity and anonymity. Too many selectable blocks allow the data processor to deanonymize the trip to that degree, thus degrading user privacy. The quantity can be defined in advance by Ψ and cannot be changed afterward.

gets a discount once an insurer accepts submitted and anonymized data. As a consequence, fraudulent behavior is not beneficial.

P1 Data has not been tampered with A trustworthy EMBEDDED-SIGNER unit processes each sensor value as soon as it is generated on a lower level of the Android Software Stack. Each \vec{x} is then hashed and signed by this entity, using its device-dependent private key d_{OS} . A user altering some values of a \vec{x} cannot create a valid signature, eventually being detectable by the VERIFIER.

P2 Data has not been modified outside the boundaries A data processor rejects data submitted by a user unless a valid signature $\sigma^P \left(\Sigma_{B'}^{OS}, H_P \left(\widehat{B'} \right) \right)$ is shown. This, in turn, is generated by the POLICYENFORCER according to the conditions of the data processor that are transparently communicated via a policy Ψ .

P3 User has produced the data A user must prove knowledge of raw data blocks for any anonymized data block on request by a VERIFIER. Although a user may use data generated by another device, this is prevented by the device-dependent key of the EMBEDDEDSIGNER that is used to sign a data block b_i (i.e. $\sigma_{b_i}^{OS}$). This is only true if there is a data processor that should be aware of a link between the mobile device of a user and the respective data.

Trust

Our pattern provides trust in four significant positions. Trust is not only relevant for the user since his data are processed, but also for a data processor, as he uses the disclosed information for premium calculation.

T1 Box only outputs user verifiable signature There is no hidden information in the POLICYENFORCER's output since it is easily comprehensible by a user using σ^P to verify the given signature, created using a known hash function H_P and user-controlled input data.

T2 Box is in a secure enclave and cannot be tampered with A TEE guarantees that sensitive processes are carried out in the protected environment. A user cannot create a signature for the POLICYENFORCER because he does not have the required cryptographic material, i.e. the secret key σ^P that is effectively protected by a TEE.

T3 User controls which data to forward The paradigm of personal-agent trust was chosen in favor of undercover-agent trust. Each piece of

information is processed in the local domain of the user. A user solely controls every data transmission, relying on a forwarding engine.

T4 User can choose freely from policy defined values A user can select any $\check{\mathcal{E}}_{cij}$ to replace an event from a trip \mathcal{B}' . The data processor accepts a value as long as the `POLICYENFORCER` verifies the integrity. It is comprehensible for a user if policy Ψ does not hold.

Evaluation

14.4

We now present an evaluation of *kUBI* that incorporates the specific example of UBI. First, we define two utility methods based on the environment: an identification and a classification method. Then, we take a detailed look at how anonymized data is crafted and assess the level of privacy provided. The section is concluded with an analysis of the achieved integrity.

Environment

14.4.1

The context in which the evaluation takes place determines two external methods. Events of a driver are assumed to be classified accordingly using an unknown method `C`. Furthermore, there exists a method that, according to the definition of Section 9.4, attempts to identify a driver and thus violates the privacy of a subject. We define that method to be the attack presented Chapter 11 that exploits two types of events (acceleration and braking). Furthermore, we assume that a specific policy Ψ that provides $\check{\mathcal{E}}_c$ of shape 3×3 , i.e. three reference events v are present for three different event classes μ , which will be used by the `ANONYMIZER`. To further remove deterministic patterns and enhance privacy, *kUBI* is allowed to choose from multiple reference events whose DTW distance is within a threshold of 10% from the original event e_c to generate its anonymized representation \hat{e}_c .

It is challenging to define a classification method `C` that is equal to an insurer's approach, as this sensitive data is not disclosed. Hence, we assume a simplistic but feasible approach to such a method based on related work such as Marina Martinez et al. [248]. A classification method in our context is a function that takes an event and assigns it to three distinct classes based on its harshness, as we illustrated previously. Harshness is defined by the velocity difference relative to the event's duration. The assignment to a class is then based on the harshness and defined static class boundaries. The classification method is used in the context of `POLICYENFORCER` and to construct $\check{\mathcal{E}}_c$. Consequently, the number of classes

Classification
method

output by the classification method equals the number of different classes of reference events $\check{\mathcal{E}}_c$ ($\mu = 3$).

Privacy-violating method

The goal of *kUBI* is to privacy-enhance existing business models and, therefore, protect a user's privacy within potentially dangerous information disclosures. To assess the ability of the framework to improve privacy, we deploy the identification method presented in Chapter 11 as it is specifically adapted to the context. It is based on kNN with DTW as its distance metric and outputs confidence, which tells how likely the trip is from a user. The final decision is made based on a maximum guess. A decrease in detection confidence is considered productive and supportive for the suitability of *kUBI* to the problem. All the illustrated assumptions hold: First, the attacker receives a stream of measurements that represent a trip \mathcal{B}' , although they have been replaced with anonymized versions $\widehat{\mathcal{B}}'$. Recall that a data block is a container for multiple measurements \mathcal{M} , hence the data format allows event extraction as presented in Section 11.3. Then, the identification, as well as the classification, is based on events \mathcal{E}'_c making them significant. Also, the set of drivers \mathcal{P} is a closed-set known in advance, and the identification approach has been trained accordingly. Lastly, a trip \mathcal{B}' always is derived from a single driver so that there are no unrealistic driver changes during a trip.

14.4.2 Anonymized Trip

The anonymization is based on replacing events with reference events. It is crucial to maintain a steady integer and sound trajectory. Reference events are normalized in $[0, 1]$, so they require rescaling when used as a replacement since every event presents another range of values.

Example

Figure 14.5 shows an excerpt from a trip of a driver. Illustrated are three braking events (e_{c0}, e_{c1}, e_{c3}) and one acceleration event (e_{c2}). The colors indicate an anonymized event created from a reference event or the event as the user's mobile device recorded it. The course of the sensor readings (shown is the velocity) is very similar, yet identification hints for a driver are removed. If the data processor offers fewer reference events as part of its policy, such as only one reference event per class, the deviation of the two curves increases for events. This is because the algorithm has fewer options to minimize the DTW distance.

14.4.3 Privacy Enhancement

The question of to what extent *kUBI* can increase the privacy of individual drivers by anonymizing trips is now discussed. Essential for providing anonymity is to

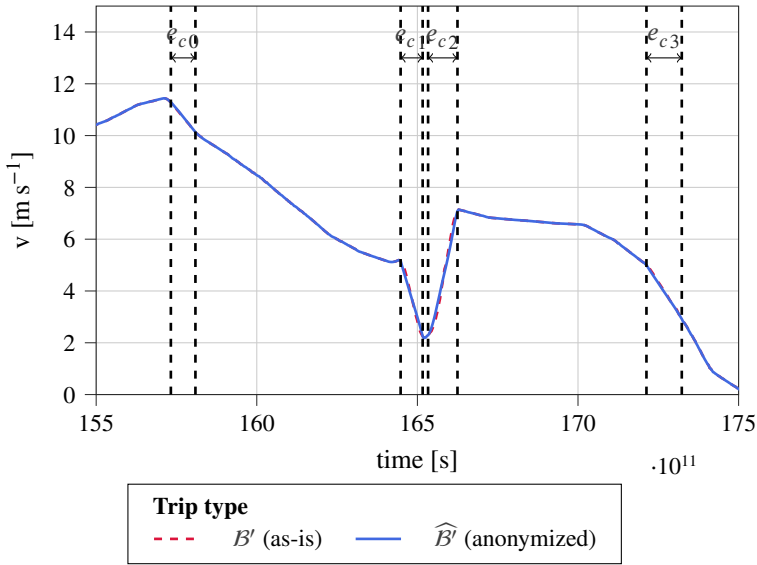


Figure 14.5

Excerpt of a trip from a driver. It shows both, the raw version B' as it is recorded by the user's device and the anonymized version \widehat{B}' after processing through $kUBI$. The course of the sensor readings (shown is the velocity) is very similar, yet identification hints for a driver are removed.

decrease the successful identification rate of the attack algorithm. Note that a successful identification occurs if a trip can be assigned to the correct driver, which happens when most of the events found in a trip represent a driver.

As we reran experiments from Section 11.4, results are comparable. In particular, confidence should be investigated initially. Although previous identification attempts showed high confidence in individual events and allowed conclusions to be made about drivers, this picture becomes less clear. Table 14.1 shows an example of the evaluation of the maximum-based approach for a trip with five events. All known drivers are available as candidates. It is striking that the confidence (or probability) across events shows identical rates for different drivers. This is because reference events are generalized events, and thus the kNN algorithm uses, as part of the voting process, i.e. the distribution of k shares to the most similar events (the minimum DTW distance), a different data basis. Typically speaking, the unknown object (gray square in Figure 11.11) is no longer close to the actual driver cluster but is arbitrarily placed in the space

Identification confidence

between the clusters of drivers. An approximation to a cluster only occurs if a driver has over proportionally many events of the class of the corresponding reference event. In contrast, Section 11.4 marks drivers or events with the same confidence whose clusters are similar. This is what *kUBI* tries to achieve since all drivers of the same type should form a cluster in the sense of *k*-anonymity.

Table 14.1 Overview of a trip evaluation to eventually predict a driver based on anonymized events as presented. The trip contains five events of the same type with five potential drivers. The prediction is based on a driver’s likelihood to be the correct driver.

Driver	Event					Pred	Pred
	1	2	3	4	5		
A	0.07	0.00	0.13	0.13	0.00	-0.67	-1.00
B	0.13	0.00	0.13	0.13	0.07	-0.53	-0.83
C	0.20	0.33	0.27	0.27	0.27	0.33	0.30
D	0.53	0.47	0.27	0.27	0.33	0.87	1.00
E	0.07	0.20	0.20	0.20	0.33	-0.00	-0.13

Analysis of
drivers

For example, the correct driver for the given example (c.f. Section 11.4) is C. However, the algorithm predicts D, mainly due to events one and two. These are overweighted, as their confidence compared to other drivers is significantly higher for driver D. While this strategy works well in the non-anonymized cases, it produces wrong results in the *kUBI* case. Other events, especially three to five, are very similar considering the confidence between drivers C and D and also between driver E. Interestingly, high confidence can be observed for missed driver predictions, higher than for correct ones. The non-normalized prediction (“*pred*”) shows uncertainty as to whether C or D is the correct driver—even E can still be considered. Exclusively, A and B do not play a role in the decision process and can be excluded from the probable drivers.

Success rate

Based on previous insights, the success rate of the anonymization approach *kUBI* is to be discussed. Summarizing results are shown in Figure 14.6 that can be compared to the ones in Section 11.4.3 ($|\mathcal{P}| = 5$). The figure shows the success rate for two cases based on the number of events found in a trip. First, the percentage of success identified trips where the algorithm was able to find the originating driver, and second, the same situation but for anonymized trips. It is imminent that the identification task becomes more manageable with more events as the success rate strongly increases above 90% in the first case. The performance has already been discussed in Section 11.4. The relationship

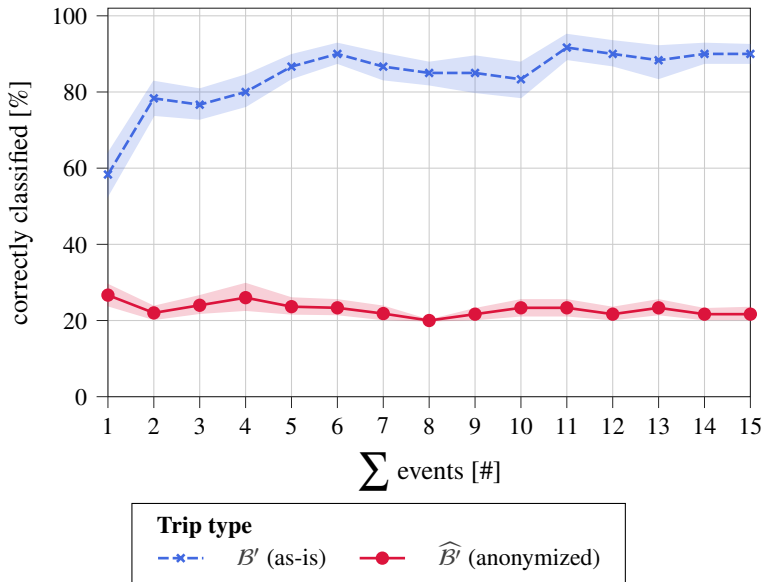


Figure 14.6

Plot of the correlation between the number of events and the correctly recognized drivers. The privacy-enhancing effect of *kUBI* becomes visible. As the number of events in a trip increases, the detection rates for the unanonymized trips increase, whereas the trips modified by *kUBI* do not allow identification.

between more events and the success rate is inverted in the second case with only around 20 % of the drivers correctly identified. This equals guessing for the five-driver scenario; hence, no learning of patterns can be applied to the problem.

Table 14.2 helps to understand the origin of why precisely 20 % were achieved. Apart from the correct identification in the case of unmodified trips (see Table 14.2a), in the other case all trips are attributed to one person (see Table 14.2b). This is surprising at first glance, as one would expect a random distribution with similar frequencies of predictions across all drivers. An assumption is that the cause is to be found, among other things, in the learning process (which takes place on unchanged events). The frequency of the events does not seem to be sufficiently satisfying as a reason since driver E shows the most events in the data set. Also, driver C possibly covers a broad spectrum in his driving behavior that reflects all three classes, and thus the reference events in the multidimensional space have a smaller distance to his (potentially extended) cluster. Hence,

Confusion of drivers

Table 14.2 Confusion matrix for driver identification after twelve trips each. Each trip has 15 acceleration and 15 braking events of different harshness. Trips are normalized based on the true label (rows, i.e. twelve trips).

a) Identification based on raw trips \mathcal{B}' (As-is). b) Identification based on trips processed by $kUBI \hat{\mathcal{B}}'$ (Anonymized).

		Predicted							Predicted				
		A	B	C	D	E			A	B	C	D	E
Actual	A	100	0	0	0	0	Actual	A	0	50	0	50	0
	B	0	50	8	0	42		B	0	0	100	0	0
	C	0	0	100	0	0		C	0	0	100	0	0
	D	0	0	0	100	0		D	0	0	92	8	0
	E	0	0	0	0	100		E	0	0	100	0	0
rates in percent						rates in percent							

with more events, the algorithm (w.r.t. the maximum-based approach) predicts driver C with meaningful confidence, except for trips from driver A that are confused with B and D. In conclusion, anonymization helps to provide privacy as no reasonable inferences are found.

14.4.4 Integrity Examination

As $kUBI$ selects the best suitable reference event as a drop-in replacement to generate k-anonymity using a minimization approach, a mixture between classes regarding classification can occur. This is related to the missing domain knowledge of the framework, as it is approach-agnostic and based solely on the policy to ensure appropriate anonymization while keeping integrity intact. This happens when the driven events are close to the class boundaries, as shown in Figure 14.7. For example, reference events (2), and (3) are similar according to the calculation of DTW and are within the threshold window but have not been selected in favor of (1). Other events, in particular, (5), and (7), are discarded, as one can see that the course does not match the raw event. It is a trade-off to accept when an insurer wants to keep his event selection algorithm confidential.

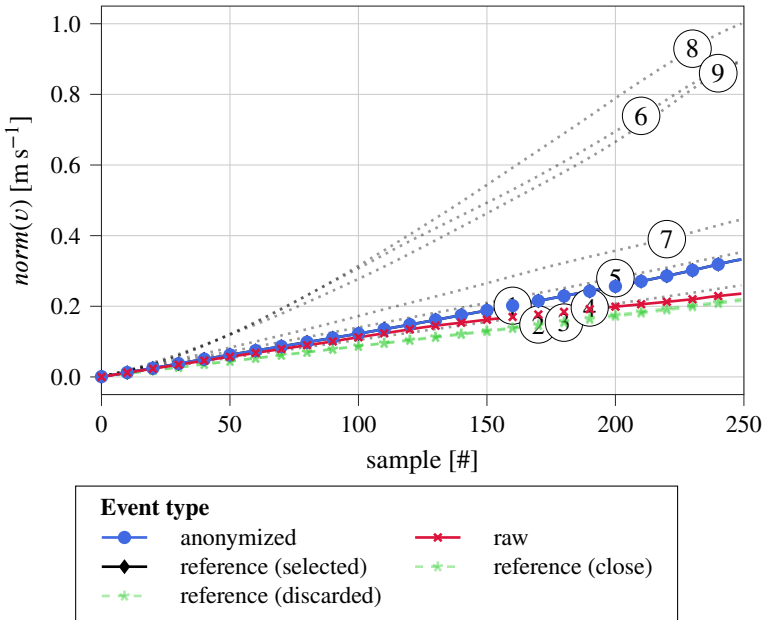


Figure 14.7

Example of the challenge to correctly select a reference event. The reference event selected by the ANONYMIZER is violating the class consistency of event replacements that is demanded by the data processor. This happens when driven events are close to class borders. It is a trade-off to take when an insurer wants to keep his event selection algorithm confidential.

However, such mismatches must be detected before reaching the data processor itself, as *kUBI* proposes a POLICYENFORCER component that can verify integrity on the user side based on the data processor’s behalf. The ANONYMIZER computes the minimum distance between an event traveled $e_c \in \mathcal{E}'$ and the various reference events $\check{\mathcal{E}}_c$ provided. Then, based on a threshold, a set of potential candidates $e_c \in \check{\mathcal{E}}_c$ is formed, and a random element is drawn. The resulting trip $\hat{\mathcal{B}}^i$ is checked by the POLICYENFORCER. In case of rejection, the random seed of the ANONYMIZER can be changed so that other candidates are randomly picked from the respective candidate set for each event in a new iteration.

POLICY-
ENFORCER

The assessment shows that the deviation of the class distribution is acceptable and requires little additional work in favor of privacy. It can be observed that

Drawbacks

critical errors, namely the deviation by two classes (aggressive becomes passive or vice versa), rarely occur. However, an exclusively minimal-based selection approach requires further optimization. *kUBI* can be suitably adapted by the policy so that the general scheme of the framework remains valid.

14.5 Conclusion

We now conclude our work on *kUBI*. First, we discuss the UBI context as a major driver for the design of the framework. The framework's ability to enable privacy is shown next, including a proposal for a reference implementation. A discussion of privacy implications follows that specifically addresses the drawbacks of *kUBI*. These drawbacks are i.a. given by design due to conflicting interests of stakeholders. Generally speaking, we want to motivate discussion and exchange between the different players.

14.5.1 The UBI Context

In Section 9.4, we asked whether user privacy is compromised by attacks such as those in Chapter 11 and Chapter 12. This question could already be answered in the affirmative, yet this type of UBI insurance shows increasing prevalence (c.f. Section 9.3). Users prefer the benefits, such as discounts on premiums or more convenience of fully automated transmission of driving data through an application.

Challenges In this respect, preventing the business model is not in line with the times, so alternative solutions must be found. Due to the particular focus on sensor data and the covert processing without knowledge of the evaluation methods, protecting user data is challenging. Certain framework conditions must be adhered to so as not to reduce the significance of the data to absurdity. Existing anonymization methods for modifying data to increase privacy, such as generalization, are not suitable [322]. Either they are ineffective because the time series are complex data structures that follow specific patterns, and changes can be leveled accordingly with suitable methods and domain knowledge. Alternatively, changes to the data are only possible within a minimal scope since the meaningfulness must not be altered under any circumstances. In the case of PHYD, a premium is calculated based on minor differences in driving behavior. For example, shifting the data (adding or subtracting values) can cause a price reduction or increase. A situation that is by no means acceptable, as our stakeholder analysis shows.

It turns out that PHYD poses much more critical requirements for a PET than PAYD does since there the known methods for anonymizing locations like mix zones, k-anonymity, or similar can be used. A solution can only be achieved through the cooperation of the different stakeholders. In addition, this is to be regarded as productive since mutual support increases confidence in the whole process. Eventually, a solution must be transparent and verifiable by external instances to be sound.

Cooperation
is key

***kUBI* as a Balance**

14.5.2

To address the challenge of processing sensitive sensor data, we present *kUBI*, a hybrid proposal that balances stakeholder interests accordingly. The framework splits data processing between the respective trust zones (User Domain and Business Domain). Data is collected in the user domain and anonymized before leaving the user domain. To maintain the integrity of the data and allow meaningful analysis, a data processor can define appropriate conditions through a policy. Control of the policy is possible for everyone so that transparency and trust are created and strengthened. *kUBI* embeds itself into the design of existing mobile OSs such as Android and is therefore retrofittable. No changes to the architecture are required, but existing components, such as a TEE (Hostile Domain), are used. *kUBI* does not require domain knowledge and, therefore, is not tied to a specific model.

Using the PHYD business model, the proposal was evaluated based on real data and the identification attack that was introduced in a previous chapter. It succeeds in establishing anonymity based on multidimensional reference events that are provided as a black box. Thus, in the experiments with authentic trips, it was shown that with unmodified raw data, it is possible to identify four out of five drivers in 100 % and one out of five drivers in 50 %. With anonymized data, users' privacy increases significantly so that in the end, no identification is possible anymore. Consequently, it is possible to remove the corresponding identifying features, but at the same time to further enable the classification of driving events. *kUBI* establishes k-anonymity in the context of the driver pool. However, it should be noted that the lack of domain knowledge simultaneously imposes a limitation on the data quality. The approach based on DTW-distances to select reference events does not always lead to optimal results. In the future, it would be conceivable to extend this module accordingly and to use more complex selection methods, for example, an ML-supported approach [242]. Since *kUBI* works *ex-post*, the run time is accordingly not of great importance, so such a proposal could be implemented.

Results

Adaptability A fundamental requirement for developing the framework has been that it can be easily implemented in existing models. This should ensure swift acceptance by stakeholders. However, no unnecessary hurdles should block or complicate the development of disruptive and agile business models based on sensor data. The following Listing 14.1 is a sample implementation of the API for interacting with sensors as defined by *kUBI* based on the reference implementation in Android (see Listing 4.1). As a result of the interchangeability of the components with static interfaces, rapid adaptation to current applications is ensured. As can be seen, all API calls are identical to those found in the Android reference implementation. This is an approach previously seen in similar proposals (e.g. [306]). However, each `SignedSensorEvent` also contains a data block ID that is the GUID. Once a data block is completed, for example, after one second, `onDataBlockComplete` is called. After obtaining the GUID, a developer may use it to map all sensor values to it. Additionally, he obtains the data block's signature $\sigma_{b_m}^{OS}$ (which is the result of H_{OS}). Applications that do not require integrity protection features may omit the optional data to make the interface work in the used fashion. Separating `onSensorChanged` and `onDataBlockComplete` enables real-time processing to continue when a new sensor value arrives, as signature generation is a bulk operation on multiple values that cannot be performed in real-time (i.e. it is not meaningful to compute signatures for single values in real-time).

14.5.3 Privacy Implications

The framework is also based on certain assumptions, which have already been discussed in this chapter. These assumptions are necessary because there is a high degree of uncertainty regarding the data evaluation process. Technical specifications or reference implementation for appropriate methods and building blocks do not exist and thus cannot be applied to the context.

User understanding There is inherently a difference in the expectations of the two stakeholders, namely the policyholder and the data processor, which is further reinforced by the reasons discussed in Section 1.2. Participation in a UBI process is an information disclosure process in which the data processor receives customer data, and the customer benefits from it, such as receiving a discount. One speaks of a second exchange[90] in contrast to a first exchange, in which a good is obtained by monetary means [146]. It can now be argued that this is precisely the point of exploitation initiated by a data processor, precisely that users place monetary aspects [40] above their concerns [415]. Furthermore, additional aspects, addressed in Section 1.2, such as convenience [255], optimistic bias [80] or cer-

```

1  val signedSensorManager = getSystemService(Context.SIGNED_SENSOR_SERVICE)
   ↪ as SignedSensorManager
2  val accelerometerSensor: Sensor? =
   ↪ sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)
3  val signedSensorListener = object: SignedSensorEventListener {
4      override fun onAccuracyChanged(sensor: Sensor, accuracy: Int) {
5          }
6      override fun onSensorChanged(event: SignedSensorEvent) {
7          val dT: Long = event.timestamp
8          val axisX: Float = event.values.get(0)
9          val axisY: Float = event.values.get(1)
10         val axisZ: Float = event.values.get(2)
11         val datablockId: UUID = event.datablockId
12         // use sensor values
13     }
14     override fun onDatablockComplete(id: UUID, signature: ByteArray) {
15     }
16 }
17 signedSensorManager.registerListener(signedSensorListener,
   ↪ accelerometerSensor, SensorManager.SENSOR_DELAY_FASTEST)

```

Snippet of the application of the privacy-enhanced `SensorManager`. A `SignedSensorManager` gives access to a `Sensor` that provides `SignedSensorEvents` with specific sensor readings and also a data block in accordance to *kUBI*. Listing 14.1

tain framing effects [29] like emphasizing advantages [407] are also relevant in this context. In particular, it is necessary to emphasize the circumstance of incomplete and asymmetric information [2] that is further presented in a non-intuitive and overwhelming way [216]. The UBI model is an abstract and highly complex business model. Additionally, users may be unaware of how data can be misappropriated.

It is questionable to what extent a secret event extraction method E_c is acceptable in the sense of GDPR, as the latter requires in GDPR Article 12 (1) that the business model and data processing be explained transparently and comprehensibly. However, the level of detail of this explanation is not defined. Technical post-implementation is not possible, at least based on the information we collected in the study (c.f. Section 9.3). Furthermore, to what extent users understand a technical explanation and how this influences their decision must be questioned and discussed. W.r.t. Acquisti [2], for example, it can be assumed to what extent

Business confidentiality and transparency

the information is understood, on the one hand, the scope is interpreted correctly, and how this would ultimately influence the decision, on the other hand. For insurers (or data processors), the challenge is to balance the obligations of GDPR Article 12 (1) with the need to maintain business confidentiality.

Summary In summary, these emerging and complex business models have far more implications than simply transmitting sensor data. At this point, it should also be made clear that users often have no choice because the business models require black-and-white thinking. Users are forced by consent to provide personal data to a provider as part of the business relationship; otherwise, participation is not possible [216], and consequently, a user will be absolutely excluded from the benefits even though he may only disagree with parts of the business terms. Therefore, *kUBI* can be used (at least temporarily) as a solution to deal with the current status quo and allow the user to act as an equal participant in the UBI process as much as possible in terms of a PET.

14.6 Outlook: Do We Need a TET Extension?

Giving a user control *kUBI* focuses on the desired properties of a PET that we introduced in the previous chapter. In fact, it gives back control to a user, enabling him to participate in the UBI business model as an equal player. *kUBI* allows to alter submitted data within his local and trusted domain before consenting to submit the data. Therefore, the proposal follows established frameworks on how to design privacy-friendly systems. We do not want to discuss the situation of a user declining to submit data (that is, in contradiction with Cavoukian [66] that a user is then excluded from a service), but we want to elaborate on the problem. A user who gives or does not give consent is based on the information disclosure decision process that was thoroughly discussed in Section 1.2 including the objectives that drive the decision (see Section 1.2.3). We have learned that this process is extensively complex, making it almost impossible (owing to bounded rationality) for the user to protect himself [350]. Consequently, the question arises whether a PET that involves user interaction is a meaningful and sustainable approach.

Education as a way out According to our understanding, education may be a solution to this dilemma because it can place a user on an equal footing with a data processor and thus correct the information mismatch mentioned by Acquisti [2]. However, such an approach also relies on the sincerity of the other stakeholders. The information advantage is mostly on a service-provider side; hence, he must provide information used to educate a user and make him aware of the consequences. We deliberately ignore the aspects already mentioned in Section 1.2.3 and focus on

what education can look like. Egan [120] gives the following statement on providing information: “*the more notifications you show to someone, the less likely that person is to apprehend or absorb any one particular notice and make informed choices about their data*” This brings back the dilemma of bounded rationality and the individual information disclosure decision process probably driven by benefits. Consequently, providing more information to a user, in general, may not be a feasible approach.

The information disclosure process is related to autonomy since people can only make sincere decisions if they can choose between multiple offerings that allow them to switch the service provider for discrepancies. In Wurmer et al. [407], we present three distinct levels of autonomy.

Comparability and autonomy

- ▶ First, *microscopic autonomy* within an exchange requires sincerity from a service provider so that a user within the framework of a second exchange⁶ can make a free decision that corresponds to his intentions and understanding of the added value. Manipulation of users can be carried out by hiding specific facts about the process [362]. As we have seen within the UBI survey in Section 9.3, a detailed and comprehensible description of the data evaluation does not take place.
- ▶ Second, *mesoscopic autonomy* can also imply that a user can choose between different services, as all offer similar competitive benefits. This is the case in the UBI context, where the benefit is typical of financial nature. However, the free choice of provider is also based on knowledge and comparability of the required data and the processing steps, so a detached consideration of the benefits is not expedient without extensive knowledge of the privacy costs.
- ▶ Third, autonomy can be transported to a *macro level* by fundamentally questioning the concept of second exchange and no longer having to trade personal data for benefits. This would eliminate the threats that have been addressed in this work. Especially with regard to Chapter 10, even seemingly “useless” data can be misused, and in this context, possibly without prior knowledge of the data collection service provider.

The concept of autonomy does not appear to be achievable through the service providers’ intrinsic arguments. On the contrary, the *macroscopic autonomy*

Transparency is not intrinsic

⁶ According to Culnan and Milberg [90], *first exchange* can be described as a transaction in which a user receives corresponding services (i.e. benefits) in exchange for money. The authors further define a *second exchange* as the exchange of data for benefits.

shows a countermovement due to the technical possibilities. The example of Usage-Based Insurance underlines this state of affairs. A way out may lie in Transparency Enhancing Technologies, which should enable “*informed consent and transparency*” [129] and thus directly support the requirements addressed by Solove [351] and in Section 1.2.1. In particular, an *ex-ante* TET (see Chapter 13) is required to support the information disclosure process, as it may be feasible to address the complex balance of benefits and costs by clarifying the hidden risks associated with shared personal data [362]. However, it is questionable on what data basis a TET can make a recommendation since here, too, there is the problem of information asymmetry due to, for example, business secrets or inadequate communication.

Ability to
provide a
recommendation

Instead, an informative tool would have to originate from the service provider, as only he has complete and comprehensive knowledge about the process. For example, when sharing an image on a social network, instead of a cryptic visibility definition, the set of all people who can see that image could be displayed. However, TETs are usually offered externally and have predefined recommendations adapted to specific situations (e.g. company, process, data subject, website) that are predefined by experts using e.g. scientific methods. However, these may be outdated, inadequate, incomplete, or inappropriate, which makes the appropriateness of a TET absurd and, in the worst case, has the opposite effect. A choice based on a TET might also have the opposite effect since a suggestion is also a source of bias in the decision-making process and must be taken into account. The technical conditions may change in the future, but a TET is based on the present conditions, according to Laufer and Wolfe [226]. Once data has been transmitted in digital form, it is typically impossible to withdraw it. Thus, if a different evaluation is conducted in the future or a new attack is identified (which is quite possible in the case of Chapter 10), the TET has retrospectively produced an incorrect suggestion on which the user relied. In the future, the credibility of a TET will be lost. Users’ faith in TETs in general may also be harmed.

Example

Let us go back to the social network example with a presentation of the group or the sharing category. A TET may indicate, as explained previously, that the group is rather large and that it may contain collaborators. Depending on the context, this may be problematic or not. Concerning our example, this means that an estimate of the privacy-criticality of the group of people is given. It is questionable on what basis this is done: the group size, for example, would be inappropriate. Imagine that a picture was taken at a company party. Hence, it is not critical if it is shared with collaborators — or is it? Consequently, the decision process is still complex, yet only the data subject can come to a conclusion.

We propose another direction of TET research to present a holistic and more adequate solution. A TET should fundamentally address how people are attuned in the sense of information disclosure and target questions about behavior to start a thinking process. As we have seen, people shorten this process, for example, due to the phenomenon of Nothing to Hide [350] or hyperbolic discounting [2]. Thus, awareness is not created by “prechewed” recommendations, but the user is encouraged to generate a personal recommendation himself. A recommendation can only be obtained subjectively. We have seen that information disclosure works with personal biases and is subjective; what sense can be attributed to a TET based on the opinions of others. Westin [399] proposes three different types of people based on their privacy behavior, namely privacy fundamentalists, privacy unconcerned, and privacy pragmatists. A TET can use the knowledge of the privacy behavior of an individual and behave like a personal assistant, giving a general notification tailored explicitly to the everyday behavior of the user but not to the website or services being interacted with. Thus, awareness can be raised without the need for precisely predefined rules that may become outdated over time. For example, a TET can try to shift groups, e.g. privacy unconcerned users may become privacy fundamentalists if they are aware of the current situation and understand the privacy relation of the data. However, the final decision is still personal but probably more thoughtful and aware. Consequently, autonomy comes from the user itself without the need for the collaboration of service providers. However, a dialogue between multiple stakeholders on an equal footing can only be beneficial, and Privacy by Design is therefore not obsolete but more relevant than ever.

Awareness
for
autonomy

Recap with Discussion and Outlook

15

The research project was motivated by the increasing adaptation of Weiser's vision of ubiquitous computing. Although it has not yet manifested itself fully, it reveals the need for an objective view of the issues involved. Mainly perceived as conducive to new types of information systems, there is an inherent threat associated with the increasing merging of the physical and virtual worlds. The complexity and sophistication of information systems are increasing significantly due to technologies such as Artificial Intelligence. However, the awareness and knowledge of people using these systems do not increase at the same rate, creating an imbalance that can lead to users being overburdened and exploited.

Remembering ubiquitous computing

Suitable methods can be used to increase awareness on the one hand, but on the other hand, the protection of users must be implemented in the systems themselves. Therefore, the central result of this research work should address both aspects. By increasing awareness of the relevance of seemingly harmless detached information such as sensor data for privacy, users should be made more sensitive. In the best case, the sensitization should not be limited to the users but also to the development of information systems themselves. Even if the paradigm of Security by Design has already been established in the development of information systems to protect the interests of providers, the concept of Privacy by Design must also be firmly established.

Guiding question

We present a summary of the work within this chapter, beginning by pointing out explicit answers to the Research Questions in Section 15.1. Next, we give a thorough condensation in Section 15.2. Finally, the dissertation is concluded with an outlook in Section 15.2.

Structure

Research Questions

15.1

The dissertation is divided into four parts, each motivated by the research questions outlined in Section 1.3.1. The following is a critical summary of the findings and answers to these questions.

1. *How can the inertial sensor data of smartphones be collected and evaluated as robustly as possible in cars?*

The research question of how sensor data based on IMU can be collected as qualitatively as possible using a smartphone cannot be answered to the last extent due to the range of identified internal and external influences that operate in the set scenario of a moving vehicle. Based on a literature review, Section 4.2 analyzed common subject approaches and created a taxonomy that describes sources of error and their effects and projects appropriate countermeasures. As a result, architecture could be developed and implemented in Section 4.3 that integrates and significantly extends existing approaches and is available as a retrofittable solution. The practical evaluation with the alternating parameters vehicle, smartphone, and track shows that the sensor data have sufficient quality for further processing, and based on IMU data, information like distance or speed can be interpolated. A side-channel attack developed (c.f. Chapter 12) exploiting this data underlines the feasibility of the proposal. However, it should be noted that it remains a challenge to generate data of consistent quality.

2. *How can smartphone sensors be used as enablers for privacy-friendly applications in traffic scenarios? What possibilities do they open up?*

The potential of user-related data in the mobile environment has been the focus of academia and industry, as crowdsourcing and crowdsensing increase the overall quality and distribution of services [55]. However, privacy-friendly processing is of particular interest, with multiple approaches identified within this work in Chapter 6 with Privacy by Design being the recommended approach. Based on the findings and generally identified constraints, two system proposals are presented that already exist in the scientific literature but lack crucial protection of users' sensitive data. In Section 6.2 and in particular in Chapters 7 and 8, approaches are presented that prove the ability to combine mutual requirements, namely integrity and privacy.

3. *What privacy risks are induced or increased by the omnipresence of smartphones in vehicles?*

To answer this research question, a Structured Literature Review was conducted in Chapter 10 that identifies five attack classes and highlights that the permission system in Android, presented in Section 3.1, is not sufficient to adequately and fully protect users. Furthermore, the side-channel attacks presented reveal that mainly zero-permission sensors are abused, which are also used in value-added services such

as Usage-Based Insurance (c.f. Chapter 9). Based on the findings, two classes of attacks are presented in Chapters 11 and 12, which are specifically intended to highlight the threats to privacy and the lack of purpose limitation of existing value-added services such as UBI that operate exclusively on data from zero-permission smartphone sensors. The presented approaches use the data available in this environment as a constraint. It is shown that the attacks undermine the user requirements and needs established in Section 1.2.

4. *What methods can be used to protect the privacy of users, especially in the Pay-How-You-Drive scenario?*

Motivated by permission systems present in modern mobile OSs (c.f. Section 3.1), it could be shown on the basis of SLR in Chapter 10 that they cannot provide comprehensive protection for technical, but also for organizational and personnel reasons. In particular, for existing business models such as UBI with information disclosure based on a disclosure process and the weighting of advantages and disadvantages (c.f. Section 1.2.3), additional technical protection measures are necessary that are not overburdening the user. Based on a comprehensive literature study, it was first possible to define the concept of Privacy Enhancing Technologies in Chapter 13 and to identify technical-organizational building blocks and establish connections with data requiring protection. Based on the requirements and problems of Chapter 9, in conjunction with the findings of PETs, a separate PET was developed that is specifically tailored to the complex and tightly specified business model UBI with its forms PAYD and PHYD. The suitability to protect personal data is shown by a comprehensive real-world evaluation based on the attack from Chapter 11. The proposal highlights the compatibility of users' privacy claims with service providers' quality requirements, and finally, together with the Privacy by Design-based proposals (c.f. Chapters 7 and 8 and Section 6.2), answers the dissertation's leading theme.

Summary of Results

15.2

The dissertation holistically addresses the complex issue of reconciling privacy with data quality (e.g. accuracy, comprehensibility, relevance, availability). Next to the guiding research questions with the answers presented in Section 15.1, a comprehensive summary is presented that discusses each topic presented in this work.

- Chapter 1 In Chapter 1, the topic is motivated, and the research proposal is illustrated with the presentation of the RQs and the methodology. The motivation is based on the fact that with ubiquitous computing and disruptive approaches such as crowdsourcing and crowdsensing, the complexity of information systems is imminent, which is hard to grasp from a privacy perspective. In addition, a comprehensive definition of privacy is given, and the complex process of information disclosure is introduced, which serves as the basis for privacy-relevant design decisions throughout this dissertation.
- Chapter 2 Chapter 2 introduces the basics of sensor data and addresses the similarity of smartphone sensors and sensors as they are installed in vehicles. Furthermore, sensors found in a smartphone are introduced in the example of Android. On this basis, the terms FPD and FCD that make use of the so-called Inertial Measurement Unit are discussed. In preparation for processing the data, sources of errors are identified that affect the quality of the measured sensor values. Appropriate approaches to detect and correct measurement errors are presented based on the literature. It is found that applied methods have to be specifically adapted to the process that handles the data.
- Chapter 3 Next, Chapter 3 introduces the Android permission system that aims to protect sensor data. The basic types of sensors are presented in terms of their level of protection, and the processes for requesting data from the user is discussed. Finally, the cat-and-mouse game between OS developers and adversaries is discussed to motivate the problems and depict holes in the permission system.
- Chapter 4 An essential point in the use of IMU is the omission of additional sensors such as the GPS for reasons of efficiency. In Chapter 4, a method is presented based on an SLR to collect extensive FPD. The limited accuracy of existing smartphone sensors and the heterogeneity of the environment are taken into account. A five-dimensional taxonomy is presented that integrates methods and approaches to generate meaningful sensor data. With *alignr*, a freely available and retrofittable Android module is presented that implements the findings and can provide proven data quality.
- Chapter 5 With the help of the previously developed application, a data set is created for empirical investigations within the scope of this work. The presentation is done in Chapter 5. Based on the collected data, first events (brakes, accelerations, turns) are addressed, representing particular patterns in the sensor data and giving insight into the environment in which the data was collected. Additionally, a notation is introduced to support further understanding of the work.

Chapter 6 specifies the information system and includes a critical assessment of the various claims of stakeholders. The concept of privacy-conscious data processing in an information system is presented on multiple levels. The evolutionary construct of Privacy by Design is introduced, as are the accompanying building blocks, methodologies, and processes for achieving data reduction. As a result, the output of the research project (and its respective artifacts) is contextualized and underlines that achieving a privacy-friendly architecture does not mean sacrificing other qualitative objectives. Chapter 6

In Chapter 7 a research project is presented that addresses the tension between integrity and confidentiality. The project is located in the context of crowdsensing and crowdsourcing, in which vehicles transmit speed information to a central entity that can derive a utilization of the road network based on the information. Our architecture follows the Privacy by Design claims and distributes knowledge among several entities of different interests. A simulation shows that dichotomy does not have to lead to a reduction in the quality and accuracy of data and that users can enjoy a high degree of privacy at the same time. Chapter 7

The potential of sensor data acquired from a smartphone is discussed in Chapter 8 with a focus on crowdsensing and crowdsourcing methods. We present how to detect complex traffic patterns such as traffic circles, traffic lights, and road works. In addition, semantic information can be provided about the road surface. The chapter intends to value the separation of processes within a complex, multi-instance information system by providing a system architecture incorporating the Privacy by Design paradigm. Due to the intelligent separation, the purpose limitation of raw data can be realized, ultimately resulting in more privacy-friendly system designs while still benefiting from ubiquitous computing. Chapter 8

Chapter 9 introduces the model of Usage-Based Insurance and related types, processes, and designs. The specifications are shown to be fundamentally based on the processing of extensive sensor data collected by smartphones and transmitted for analysis. The approaches of Privacy by Design are not evident, and the evaluation is a nontransparent black-box process without technical protection measures regarding the aspects of data minimization and purpose limitation as defined by GDPR. Consequently, the question of the protection of the privacy of the users when participating in such a model is raised. Chapter 9

Side-channel attacks are an omnipresent risk in the mobile environment and especially with sensor data, as shown by a conducted SLR in Chapter 10. The literature study allows us to divide attacks into five classes and to identify the dominant attack vectors in the form of sensors. These can be put in relation Chapter 10

to Android's permission system from Section 3.1 and show that attacks based on zero-permission sensors are widespread. In conclusion, it can be noted that mitigation of the attacks is challenging and hardly relevant in the scientific literature so far; Android's permission system itself does not provide adequate protection.

- Chapter 11 An identified threat to privacy in the previous chapter is fingerprinting users based on sensor data. In Chapter 11, an ML-based identification approach is presented that settles in the UBI environment, i.e. is restricted to sensor data from the accelerometer and gyroscope. A conducted SLR presents different methods and processes to perform identification; however, none of them is feasible for the given setting due to various properties. Based on the data set introduced in Chapter 5, an empirical evaluation was performed that provides reliable identification rates for large sets of drivers with limited available data, further highlighting the threat to privacy even for irregular drivers.
- Chapter 12 The SLR performed in Chapter 10 presents another primary attack class, namely location inference. In Chapter 12, we present a location inference attack that again is settled and constrained to the specifications of UBI. A SLR yields multiple location-inference attacks; however, they are limited in the required knowledge, area size, run time, or sensor data. Consequently, the presented attack addresses the said drawbacks and extends the research in this area. The attack shows that even without access to GNSS, an adversary (or service provider) is capable of tracking driven routes for large areas, allowing him to generate comprehensive movement profiles. The data set presented in Chapter 5 is used again for real-world evaluation.
- Chapter 13 Within Chapter 13, methods, concepts, and techniques to protect a user's privacy in existing information systems and use-cases are introduced. We formally define the terms PETs and TETs and delimit them from each other. A SLR gives answers on current PETs including their architecture and building blocks, although approaches are primarily user-focused, i.e. a user needs to assess the sensitivity of data in transit. W.r.t. Section 1.2 no satisfactory solution capable of protecting users can be identified even in the complex UBI use case.
- Chapter 14 Motivated by the lack of adequate protection mechanisms in the UBI setting, a framework is presented in Chapter 14 that integrates the knowledge of Chapters 3, 10 and 13 and Section 1.2. We present a new anonymization model for sensor data, as existing methods are not feasible [322]. An evaluation based on the real world data set (c.f. Chapter 5) incorporates the attack from Chapter 10 but also considers the UBI constraints from Chapter 9. The privacy among a set

of drivers can be ensured while still allowing a meaningful assessment of the PHYD features.

Discussion and Outlook

15.3

We conclude this work with a recap of the still existing gaps and an outlook of future research directions based on these gaps.

Even though each part and chapter discusses a single topic in an isolated fashion, they, in summary, answer the guiding question of this work: *How to design privacy-friendly information systems addressing humans while taking benefit of the new technologies within the ubiquitous computing paradigm?* However, this question cannot be answered within this research project's limited scope and resources. The question should rather be phrased "*can dual use of sensor data be prohibited or prevented the way it works today?*". W.r.t. the knowledge we gained during this project, the answer will probably be that this ambitious goal may not be reached the way it works today. Technology can be designed to protect technology from technology, but technology at the moment is not designed to protect users from technology. We have seen that sophisticated Privacy by Design architectures and powerful PETs exist, capable of protecting sensitive user data from most threats and entities, sometimes even including the service provider itself. It is still assumed that the user is mature and can make beneficial and personal decisions, although it was shown that such a process is almost impossible due to various factors. Although this is the basis of most PETs and business models in which users participate, technology cannot relieve a user from deciding to disclose information – it can only support him with convenience, education, and even protection [332]. Furthermore, even in some situations, technology is still unable to protect itself from technology (i.e. when subjected to side-channel attacks), further stressing the fact that a user must be aware of the sensitivity of his data and the consequences of disclosure.

Technology protects technology, not humans

Put aside that most PETs are tailored for specific use cases or environments, PETs are considered personal assistants to a user, running in a trusted environment for the user's benefit. Therefore, their output has a high impact on the information disclosure process. Related to this topic is Privacy by Design, as it can also increase the trustworthiness of an information system similarly depending on the entities involved. The challenge is to gain the trust of users, regardless of how. Trust may be defined as the "*principal having belief and confidence in the goodwill of the agent's intent and behavior, and thus being willing to risk exposure to vulnerability or take the leap of faith*" [148]. This step is essential in

PETs require trust

a user-service relationship and may be a steady and long-lasting process. Once trust has been established with a provider, it can be assumed that users will also have confidence in the services it offers or, in contrast, hesitate or avert to use it if they distrust the provider.

Split trust
continuum

Trust and distrust seem to be directly related, but the so-called split trust continuum breaks with this convention [363]. The construct describes a situation where trust and distrust are distinct aspects. Therefore, a lack of trust in a service provider does not necessarily imply a presence of distrust. Both can be lacking at the same time. It is crucial to understand that a user may not distrust an entity but still does not put significant amounts of trust in it. This paradoxical condition can be the basis for developing trustworthy information systems to achieve a situation that corresponds to the definition of trust mentioned earlier. Users may distrust a service provider because he wants to earn money with the users' data, the so-called second exchange repeatedly outlined in this work. Recall that a user's personal assistant is located in his trusted domain which may be his smartphone that collects all data (refer to the data minimization strategies in Section 6.1.2). Therefore, a user may have reliance on the predictability of the smartphone OS and the respective hardware (i.e. lack of distrust). At the same time, he ignores whether, for instance, Google is trustworthy or not (i.e. lack of trust). Consequently, if the personal assistant is placed in a domain that is not subject to distrust but at the same time does not require any trust, it can function as a proxy between the service provider and the user itself (somehow related to the Trusted Third Party (c.f. Chapter 6) but without the need for trust).

Split trust
example

We know such systems from COVID-19 notification apps that warn users if they were exposed to COVID-19 cases. The OS provides functionality in the form of the `Exposure Notifications API` [103] and a national companion app provided e.g. by the German government. The sensitive task of tracking locations and contacts is performed by the OS itself without the need to transmit any trajectories. Hence, the information is kept confidential within the *not-distrusted*, *not-trusted* local domain of a user. The smartphone can decide to inform a user of COVID-19 exposures based on the information it receives from a central entity. The entity itself never needs to know which contacts were met. Thus, the German Corona Warn-App is an example of a Privacy by Design (see [85] for complete documentation).

Further
research

This presented approach makes personal processing of sensor data possible in a privacy-friendly manner. Sensitive raw data can still be secured by aggregating and delivering results in a meaningful manner. For example, Android includes a sensor called `Sensor.TYPE_STEP_DETECTOR` that is triggered whenever a

user takes a step. Data aggregation eliminates the need to derive data from the raw accelerometer and inhibits evaluation in the sense of a side-channel assault. No local attacks or data outflows are possible because protection is available immediately in the local domain. In the future, it is possible to improve this technique and provide general fundamental functions directly from the device or from the OS itself. Concerning the realm of PHYD, for example, evaluations of driving characteristics could occur in the device and be regulated by parameters supplied by insurance companies. Corona's warning app operates similarly. The set of parameters for weighing positive and negative situations is provided by a central entity in the form of standardized files and is evaluated locally by the app. This work should catalyze further research on methods other than centralized review in closed systems to mitigate the risk of a data leak (intentional or unintentional).

Of course, these technical approaches do not negate the importance of user education as a core principle. However, the development of TETs should focus on user education as a whole in favor of providing specific recommendations for specific steps on a particular website or system. In fact, TETs should try to stimulate the user's decision-making process, liberating him from his entrenched way of thinking. Only a reflective, inclusive, and educated society operating at eye level is capable of averting conditions akin to those described in Orwell's 1984.

Appendix

List of Functions

A

In the context of this work, some (auxiliary) functions are defined. The following is a list of all functions classified by their area.

Statistics

- O Count the occurrences of an object in a series of values
- idx Get the position of an element in an ordered structure such as a sequence (i.e. the index)
- max Get the maximum value of a digit-based series of values
- mean Get the mean of a digit-based series of values
- med Get the median of a digit-based series of values
- min Get the minimum value of a digit-based series of values
- sign Get the sign of a number (plus or minus)
- std Calculate the standard deviation of a list

Measurements

- E_a Find atomic events in a time-series
- E_c Find complex events in a time-series
- R Resample a time-series to a specific frequency or length
- conv Convert sensor readings between specific formats (e.g. data block to measurements)
- dur Calculate the total duration of a time-series (e.g. sequences of measurements)
- norm Normalize a value between a defined range (default [0, 1])

Cryptographic

- DecA Asymmetrically decrypt a payload using a d
- EncA Asymmetrically encrypt a payload using a c
- Enc Symmetrically encrypt a payload using a key
- H Generate a unique hash value for supported structure

SigC Create a signature for a given payload using d

SigV Verify a signature for a given payload using c

GIS

A Calculate the angle between two supported structures (e.g. location pairs)

B Get the heading direction for a supporting structure (e.g. OSM ways)

C Get the curvature of a supported structure (e.g. OSM ways)

Dist Calculate the distance of two measurements based on the respective *loc*

L Get the total length of a supported structure (e.g. OSM ways)

P Create a path shape from a sequence of measurements

v Calculate the velocity for two measurements

Time-Series

C Assign an event to a non-overlapping category

M_w Convert a measurement to a specific type to be used by W

W Generate sliding windows of length ω

rfw Calculate the reference window

Structured Literature Review

B

Within the scope of this work, several SLR were performed. Their methods have been presented and explained in detail at the relevant place.

Literature was usually extracted from literature databases. For this purpose, appropriate search strings were designed and applied accordingly, taking into account database-dependent syntax. For the presentation of the syntax, the following notation (represented in extended Backus-Naur form) is used in the context of this work.

A word on search string notation

```
<char> ::= 'A' | 'B' | '...' | 'Z' ;
<num>  ::= '0' | '1' | '...' | '9' ;
<term> ::= ( <char> | <num> ) ,
          { ( <char> | <num> | ' ' ) } ,
          [ '*' ] ;

<or>   ::= '|' ;
<and>  ::= '&' ;
<concat> ::= <or> | <and> ;
<invert> ::= '~' ;

<query> ::= [ <invert> ] ,
           '(' ,
           <term> ,
           { <concat> , ( <term> | <query> ) } ,
           ')' ;
```

The notation allows the concatenation of several expressions, which are linked by AND or OR. A ~negates an expression and serves to exclude the terms. Combinations of words can be combined arbitrarily, where a * corresponds to the classic wildcard parameter for any string at this point (e.g. *attack** will match *attack*, *attacking*, *attacker*, and so on). It will be interpreted depending on the database where the search is performed.

References

- [1] A. Acquisti and J. Grossklags. “Privacy and rationality in individual decision making”. In: *IEEE Security and Privacy Magazine* 3.1 (2005), pp. 26–33. DOI: 10.1109/msp.2005.22.
- [2] Alessandro Acquisti. “Privacy in electronic commerce and the economics of immediate gratification”. In: *Proceedings of the 5th ACM conference on Electronic commerce - EC '04*. Ed. by Jack Breese, Joan Feigenbaum, and Margo Seltzer. New York, New York, USA: ACM Press, 2004, p. 21. DOI: 10.1145/988772.988777.
- [3] Alessandro Acquisti, Curtis Taylor, and Liad Wagman. “The Economics of Privacy”. In: *Journal of Economic Literature* 54.2 (2016), pp. 442–492. DOI: 10.1257/jel.54.2.442.
- [4] Zahra Aivazpour and V. Srinivasan Rao. “Impulsivity and Information Disclosure: Implications for Privacy Paradox”. In: *Proceedings of the 52nd Hawaii International Conference on System Sciences*. Ed. by Tung Bui. Proceedings of the Annual Hawaii International Conference on System Sciences. Hawaii International Conference on System Sciences, 2019. DOI: 10.24251/hicss.2019.586.
- [5] Icek Ajzen. “The theory of planned behavior”. In: *Organizational Behavior and Human Decision Processes* 50.2 (1991), pp. 179–211. DOI: 10.1016/0749-5978(91)90020-t.
- [6] G. Akerlof. “The Market for “Lemons”: Quality Uncertainty and the Market Mechanism”. In: *Essential readings in economics*. Ed. by Saul Estrin. Basingstoke: Macmillan, 1995, pp. 175–188. DOI: 10.1007/978-1-349-24002-9_9.
- [7] Pawel Aksamit and Marek Szmechta. “Distributed, mobile, social system for road surface defects detection”. In: *2011 5th International Symposium on Computational Intelligence and Intelligent Informatics (ISCIII 2011)*. Piscataway, NJ: IEEE, 2011, pp. 37–40. DOI: 10.1109/isciii.2011.6069738.
- [8] Muhammad Alam, Joaquim Ferreira, and José Fonseca. *Intelligent Transportation Systems*. Vol. 52. Cham: Springer International Publishing, 2016. DOI: 10.1007/978-3-319-28183-4.
- [9] Efthimios Alepis and Constantinos Patsakis. “Unravelling Security Issues of Runtime Permissions in Android”. In: *Journal of Hardware and Systems Security* 3.1 (2019), pp. 45–63. DOI: 10.1007/s41635-018-0053-2. URL: <https://link.springer.com/article/10.1007/s41635-018-0053-2>.

- [10] Aya Hamdy Ali, Ayman Atia, and Mostafa-Sami M. Mostafa. "Recognizing Driving Behavior and Road Anomaly Using Smartphone Sensors". In: *Multigenerational online behavior and media use*. Ed. by Information Resources Management Association. Hershey, PA: IGI Global, 2019, pp. 651–667. DOI: 10.4018/978-1-5225-7909-0.ch036.
- [11] Hadiyattullahi Tanko Aliyu and Yogachandran Rahulamathavan. "Type and Leak Your Ethnicity on Smartphones". In: *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 2557–2561. DOI: 10.1109/icassp.2019.8682718.
- [12] Azza Allouch et al. "RoadSense: Smartphone Application to Estimate Road Conditions Using Accelerometer and Gyroscope". In: *IEEE Sensors Journal* 17.13 (2017), pp. 4231–4238. DOI: 10.1109/jksen.2017.2702739.
- [13] Iman M. Almomani and Aala Al Khayer. "A Comprehensive Analysis of the Android Permissions System". In: *IEEE Access* 8 (2020), pp. 216671–216688. DOI: 10.1109/access.2020.3041432.
- [14] Aljawharah Alnasser, Hongjian Sun, and Jing Jiang. "Cyber security challenges and solutions for V2X communications: A survey". In: *Computer Networks* 151 (2019), pp. 52–67. DOI: 10.1016/j.comnet.2018.12.018.
- [15] Yazan A. Alqudah and Belal H. Sababha. "On the analysis of road surface conditions using embedded smartphone sensors". In: *2017 8th International Conference on Information and Communication Systems (ICICS)*. Piscataway, NJ: IEEE, 2017, pp. 177–181. DOI: 10.1109/iacs.2017.7921967.
- [16] N. S. Altman. "An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression". In: *The American Statistician* 46.3 (1992), pp. 175–185. DOI: 10.1080/00031305.1992.10475879.
- [17] Heba Aly, Anas Basalamah, and Moustafa Youssef. "Automatic Rich Map Semantics Identification Through Smartphone-Based Crowd-Sensing". In: *IEEE Transactions on Mobile Computing* 16.10 (2017), pp. 2712–2725.
- [18] Mohamed Akram Ameddah, Bhaskar Das, and Jalal Almhana. "Cloud-Assisted Real-Time Road Condition Monitoring System for Vehicles". In: *2018 IEEE Global Communications Conference (GLOBECOM)*. Piscataway, NJ: IEEE, 2018, pp. 1–6. DOI: 10.1109/glocom.2018.8647334.
- [19] B. Y. Amirgaliyev, K. K. Kuvatov, and Z. Y. Baibatyr. "Road condition analysis using 3-axis accelerometer and GPS sensors". In: *Application of Information and Communication Technologies - AICT2016*. Piscataway, NJ: IEEE, 2016, pp. 1–5. DOI: 10.1109/icaict.2016.7991662.
- [20] S. Abhishek Anand and Nitesh Saxena. "Coresident evil". In: *Proceedings of the 10th ACM Conference on Security and Privacy in Wireless and Mobile Networks*. Ed. by Guevara Noubir, Mauro Conti, and Sneha Kumar Kasera. New York, NY, USA: Acm, 2017, pp. 173–183. DOI: 10.1145/3098243.3098256.

- [21] S. Abhishek Anand and Nitesh Saxena. “Speechless: Analyzing the Threat to Speech Privacy from Smartphone Motion Sensors”. In: *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018, pp. 1000–1017. DOI: 10.1109/sp.2018.00004.
- [22] S. Abhishek Anand, Prakash Shrestha, and Nitesh Saxena. “Bad Sounds Good Sounds: Attacking and Defending Tap-Based Rhythmic Passwords Using Acoustic Signals”. In: *Cryptology and Network Security*. Ed. by Michael Reiter and David Naccache. Vol. 9476. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 95–110. DOI: 10.1007/978-3-319-26823-1_7.
- [23] Android Open Source Project. *Sensor stack*. 2020. URL: <https://source.android.com/devices/sensors/sensor-stack>.
- [24] Vittorio Astarita et al. “A Mobile Application for Road Surface Quality Control: UNIQUALroad”. In: *Procedia - Social and Behavioral Sciences* 54 (2012), pp. 1135–1144. DOI: 10.1016/j.sbspro.2012.09.828.
- [25] Vittorio Astarita et al. “New methodology for the identification of road surface anomalies”. In: *2014 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI 2014)*. Piscataway, NJ: IEEE, 2014, pp. 149–154. DOI: 10.1109/soli.2014.6960710.
- [26] Adam J. Aviv et al. “Practicality of accelerometer side channels on smartphones”. In: *Proceedings of the 28th Annual Computer Security Applications Conference on - ACSAC '12*. Ed. by Robert H’obbes’ Zakon. New York, New York, USA: ACM Press, 2012, p. 41. DOI: 10.1145/2420950.2420957.
- [27] Zhongjie Ba et al. “Accelerometer-based smartphone eavesdropping”. In: *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. New York, NY, USA: Acm, 2020, pp. 1–2. DOI: 10.1145/3372224.3417323.
- [28] Marcin Badurowicz, Jerzy Montusiewicz, and Pawel Karczmarek. “Detection of Road Artefacts Using Fuzzy Adaptive Thresholding”. In: *2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2020, pp. 1–8. DOI: 10.1109/fuzz48607.2020.9177822.
- [29] Young Min Baek. “Solving the privacy paradox: A counter-argument experimental approach”. In: *Computers in Human Behavior* 38 (2014), pp. 33–42. DOI: 10.1016/j.chb.2014.05.006.
- [30] Shahrzad Moeiniyan Bagheri et al. “Cost-effective ubiquitous method for motor vehicle speed estimation using smartphones”. In: *IET Wireless Sensor Systems* 8.6 (2018), pp. 340–349. DOI: 10.1049/iet-wss.2018.5023.
- [31] Xiaolong Bai, Jie Yin, and Yu-Ping Wang. “Sensor Guardian: prevent privacy inference on Android sensors”. In: *EURASIP Journal on Information Security* 2017.1 (2017). DOI: 10.1186/s13635-017-0061-8.

- [32] M. Bando et al. “Structure stability of congestion in traffic dynamics”. In: *Japan Journal of Industrial and Applied Mathematics* 11.2 (1994), pp. 203–223. DOI: 10.1007/bf03167222.
- [33] Paul Bankhead. *Reminder SMS/Call Log Policy Changes*. Ed. by Android Developers Blog: 2019. URL: <https://android-developers.googleblog.com/2019/01/reminder-smscall-log-policy-changes.html>.
- [34] Gaurav Bansal, Fatemeh Mariam Zahedi, and David Gefen. “Do context and personality matter? Trust and privacy concerns in disclosing private information online”. In: *Information & Management* 53.1 (2016), pp. 1–21. DOI: 10.1016/j.im.2015.08.001.
- [35] Susanne Barth and Menno D.T. de Jong. “The privacy paradox – Investigating discrepancies between expressed privacy concerns and actual online behavior – A systematic literature review”. In: *Telematics and Informatics* 34.7 (2017), pp. 1038–1058. DOI: 10.1016/j.tele.2017.04.013.
- [36] Davide B. Bartolini, Philipp Miedl, and Lothar Thiele. “On the capacity of thermal covert channels in multicores”. In: *Proceedings of the Eleventh European Conference on Computer Systems*. Ed. by Cristian Cadar et al. New York, NY, USA: Acm, 2016, pp. 1–16. DOI: 10.1145/2901318.2901322.
- [37] Akanksh Basavaraju et al. “A Machine Learning Approach to Road Surface Anomaly Assessment Using Smartphone Sensors”. In: *IEEE Sensors Journal* 20.5 (2020), pp. 2635–2647. DOI: 10.1109/jsen.2019.2952857.
- [38] Olivier Baudron et al. “Practical multi-candidate election system”. In: *Proceedings of the twentieth annual ACM symposium on Principles of distributed computing - PODC '01*. Ed. by Ajay Kshemkalyani and Nir Shavit. New York, New York, USA: ACM Press, 2001, pp. 274–283. DOI: 10.1145/383962.384044.
- [39] Felix Bauer et al. *Machine Learning und die Transparenzanforderungen der DS-GVO*. Ed. by Bitkom. 2018. URL: <https://www.bitkom.org/Bitkom/Publikationen/Machine-Learning-und-die-Transparenzanforderungen-der-DS-GVO.html>.
- [40] Volker Benndorf and Hans–Theo Normann. “The Willingness to Sell Personal Data”. In: *The Scandinavian Journal of Economics* 120.4 (2018), pp. 1260–1278. DOI: 10.1111/sjoe.12247.
- [41] David Berend, Shivam Bhasin, and Bernhard Jungk. “There Goes Your PIN”. In: *Proceedings of the 13th International Conference on Availability, Reliability and Security*. New York, NY, USA: Acm, 2018, pp. 1–10. DOI: 10.1145/3230833.3232826.
- [42] A. R. Beresford and F. Stajano. “Mix zones: user privacy in location-aware services”. In: *IEEE Annual Conference on Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second*. IEEE, 2004, pp. 127–131. DOI: 10.1109/percomw.2004.1276918.

- [43] Mario Luca Bernardi et al. “Driver Identification: a Time Series Classification Approach”. In: *2018 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2018, pp. 1–7. DOI: 10.1109/ijcnn.2018.8489426.
- [44] Sourav Bhattacharya et al. “Robust and Energy-Efficient Trajectory Tracking for Mobile Devices”. In: *IEEE Transactions on Mobile Computing* 14.2 (2015), pp. 430–443. DOI: 10.1109/tmc.2014.2318712.
- [45] Sebastian Biedermann, Stefan Katzenbeisser, and Jakub Szefer. “Hard Drive Side-Channel Attacks Using Smartphone Magnetic Field Sensors”. In: *Financial Cryptography and Data Security*. Ed. by Rainer Böhme and Tatsuaki Okamoto. Vol. 8975. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 489–496. DOI: 10.1007/978-3-662-47854-7_30.
- [46] Debmalya Biswas and Krishnamurthy Vidyasankar. “Privacy preserving and transactional advertising for mobile services”. In: *Computing* 96.7 (2014), pp. 613–630. DOI: 10.1007/s00607-013-0332-2.
- [47] Kenneth Block and Guevara Noubir. “My Magnetometer Is Telling You Where I’ve Been?”. In: *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. Ed. by Panos Papadimitratos, Kevin Butler, and Christina Pöpper. New York, NY, USA: Acm, 2018, pp. 260–270. DOI: 10.1145/3212480.3212502.
- [48] Geoff Boeing. “Urban spatial order: street network orientation, configuration, and entropy”. In: *Applied Network Science* 4.1 (2019). DOI: 10.1007/s41109-019-0189-1.
- [49] Nadine Bol et al. “Understanding the Effects of Personalization as a Privacy Calculus: Analyzing Self-Disclosure Across Health, News, and Commerce Contexts†”. In: *Journal of Computer-Mediated Communication* 23.6 (2018), pp. 370–388. DOI: 10.1093/jcmc/zmy020.
- [50] Rachel Botsman. “Big data meets Big Brother as China moves to rate its citizens: The Chinese government plans to launch its Social Credit System in 2020. The aim? To judge the trustworthiness – or otherwise – of its 1.3 billion residents”. In: *Wired* (2017). URL: <https://teomeuk.s3.amazonaws.com/wp-content/uploads/2017/11/05/Wired%5C%5FBig%5C%5Fdata%5C%5Fmeets%5C%5FBig%5C%5FBrother%5C%5Fas%5C%5FChina%5C%5Fmoves%5C%5Fto%5C%5Frate%5C%5Fits%5C%5Fcitizens.pdf>.
- [51] Daren C. Brabham. “Crowdsourcing as a Model for Problem Solving”. In: *Convergence: The International Journal of Research into New Media Technologies* 14.1 (2008), pp. 75–90. DOI: 10.1177/1354856507084420.
- [52] Benedikt Brecht et al. “A Security Credential Management System for V2X Communications”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.12 (2018), pp. 3850–3871. DOI: 10.1109/tits.2018.2797529.

- [53] Susanne Breitenberger, Bernhard Grüber, and Martina Neuherz. “Extended Floating Car Data - Potenziale fuer die Verkehrsinformation und notwendige Durchdringungsraten / Extended Floating Car Data - Potential for Traffic Information and required Floating Car Penetration”. In: *StraSsenverkehrstechnik* 48 (2004), pp. 522–531. URL: <https://trid.trb.org/view/944222>.
- [54] Manfred Broy and Ralf Steinbrüggen. *Modellbildung in der Informatik*. Xpert.press. Berlin, Heidelberg and s.l.: Springer Berlin Heidelberg, 2004. DOI: 10.1007/978-3-642-18732-2.
- [55] Ricardo Buettner. “A Systematic Literature Review of Crowdsourcing Research from a Human Resource Management Perspective”. In: *2015 48th Hawaii International Conference on System Sciences*. IEEE, 2015, pp. 4609–4618. DOI: 10.1109/hicss.2015.549.
- [56] Sven Bugiel, Stephan Heuser, and Ahmad-Reza Sadeghi. “Flexible and Fine-Grained Mandatory Access Control on Android for Diverse Security and Privacy Policies”. In: *Proceedings of the 22nd USENIX Conference on Security*. Sec’13. Usa: USENIX Association, 2013, pp. 131–146.
- [57] Statistisches Bundesamt. “Anteile Der Fehlverhalten Von Fahrzeugführern Bei StraSsenverkehrsunfällen Mit Personenschaden In Deutschland Von 2010 Bis 2020”. de. In: *Statista, Statista GmbH* (July 7, 2021). URL: <https://de.statista.com/statistik/daten/studie/2110/umfrage/fehlverhalten-von-fahrzeugfuehrern-mit-unfallfolge/>.
- [58] Angela Burton et al. “Driver identification and authentication with active behavior modeling”. In: *2016 12th International Conference on Network and Service Management (CNSM)*. IEEE, Oct. 2016, pp. 388–393. DOI: 10.1109/cnsm.2016.7818453. URL: <https://doi.org/10.1109%5C%2Fcnsm.2016.7818453>.
- [59] Frederico Soares Cabral et al. “An Automatic Survey System for Paved and Unpaved Road Classification and Road Anomaly Detection using Smartphone Sensor”. In: *2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*. IEEE, 2018, pp. 65–70. DOI: 10.1109/soli.2018.8476788.
- [60] Liang Cai and Hao Chen. “On the Practicality of Motion Based Keystroke Inference Attack”. In: *Trust and Trustworthy Computing*. Ed. by David Hutchison et al. Vol. 7344. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 273–290. DOI: 10.1007/978-3-642-30921-2_16.
- [61] Eyüp S. Canlar et al. “Sense-And-Trace: A Privacy Preserving Distributed Geolocation Tracking System”. In: *Security Protocols XX*. Ed. by David Hutchison et al. Vol. 7622. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 199–213. DOI: 10.1007/978-3-642-35694-0_22.
- [62] Justin Cappos et al. “BlurSense: Dynamic fine-grained access control for smartphone privacy”. In: *2014 IEEE Sensors Applications Symposium (SAS)*. IEEE, 2014, pp. 329–332. DOI: 10.1109/sas.2014.6798970.

- [63] Giuseppe Cardone et al. “ParticipAct: A Large-Scale Crowdsensing Platform”. In: *IEEE Transactions on Emerging Topics in Computing* 4.1 (2016), pp. 21–32. DOI: 10.1109/tetc.2015.2433835.
- [64] M. Ricardo Carlos et al. “Becoming Smarter at Characterizing Potholes and Speed Bumps from Smartphone Data — Introducing a Second-Generation Inference Problem”. In: *IEEE Transactions on Mobile Computing* 20.2 (2021), pp. 366–376. DOI: 10.1109/tmc.2019.2947443.
- [65] Manuel Ricardo Carlos et al. “Evaluation of Detection Approaches for Road Anomalies Based on Accelerometer Readings—Addressing <italic>Who’s Who</italic>”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.10 (2018), pp. 3334–3343. DOI: 10.1109/tits.2017.2773084.
- [66] Ann Cavoukian. *Privacy by Design - The 7 Foundational Principles*. Ontario, Canada, 2011.
- [67] Supriyo Chakraborty, Wentao Ouyang, and Mani Srivastava. “LightSpy: Optical eavesdropping on displays using light sensors on mobile devices”. In: *2017 IEEE International Conference on Big Data (Big Data)*. IEEE, 2017, pp. 2980–2989. DOI: 10.1109/BigData.2017.8258268.
- [68] Supriyo Chakraborty et al. “A framework for context-aware privacy of sensor data on mobile systems”. In: *Proceedings of the 14th Workshop on Mobile Computing Systems and Applications - HotMobile '13*. Ed. by Sharad Agarwal and Alexander Varshavsky. New York, New York, USA: ACM Press, 2013, p. 1. DOI: 10.1145/2444776.2444791.
- [69] Rosalie Chan. “Cambridge Analytica whistleblower on how the firm used Facebook data”. In: *Insider* (2019). URL: <https://www.businessinsider.com/cambridge-analytica-whistleblower-christopher-wylie-facebook-data-2019-10>.
- [70] Gayathri Chandrasekaran et al. “Vehicular speed estimation using received signal strength from mobile phones”. In: *Proceedings of the 12th ACM international conference on Ubiquitous computing*. Ed. by Jakob E. Bardram et al. New York, NY, USA: Acm, 2010, pp. 237–240. DOI: 10.1145/1864349.1864386.
- [71] Girish Chandrashekar and Ferat Sahin. “A survey on feature selection methods”. In: *Computers & Electrical Engineering* 40.1 (2014), pp. 16–28. DOI: 10.1016/j.compeleceng.2013.11.024.
- [72] David Chaum. “Blind Signatures for Untraceable Payments”. In: *Advances in Cryptology*. Ed. by David Chaum, Ronald L. Rivest, and Alan T. Sherman. Boston, MA: Springer US, 1983, pp. 199–203. DOI: 10.1007/978-1-4757-0602-4_18.
- [73] David Chaum. “The dining cryptographers problem: Unconditional sender and recipient untraceability”. In: *Journal of Cryptology* 1.1 (1988), pp. 65–75. DOI: 10.1007/bf00206326.

- [74] David Chaum and Eugène van Heyst. “Group Signatures”. In: *Advances in Cryptology — EUROCRYPT '91*. Ed. by Gerhard Goos, Juris Hartmanis, and Donald W. Davies. Vol. 547. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 257–265. DOI: 10.1007/3-540-46416-6_22.
- [75] David L. Chaum. “Untraceable electronic mail, return addresses, and digital pseudonyms”. In: *Communications of the ACM* 24.2 (1981), pp. 84–90. DOI: 10.1145/358549.358563.
- [76] Xin Chen et al. “Real-location Reporting Based Differential Privacy Trajectory Protection for Mobile Crowdsensing”. In: *2019 5th International Conference on Big Data Computing and Communications (BIGCOM)*. IEEE, 2019, pp. 142–150. DOI: 10.1109/bigcom.2019.00029.
- [77] Yinsheng Chen et al. “Grey bootstrap method for data validation and dynamic uncertainty estimation of self-validating multifunctional sensors”. In: *Chemometrics and Intelligent Laboratory Systems* 146 (2015), pp. 63–76. DOI: 10.1016/j.chemolab.2015.05.003.
- [78] Yongxi Chen and Anne S. Y. Cheung. “The Transparent Self Under Big Data Profiling: Privacy and Chinese Legislation on the Social Credit System”. In: *SSRN Electronic Journal* (2017). DOI: 10.2139/ssrn.2992537. URL: <https://core.ac.uk/download/pdf/84930493.pdf>.
- [79] Peng Cheng et al. “SonarSnoop: active acoustic side-channel attacks”. In: *International Journal of Information Security* 19.2 (2020), pp. 213–228. DOI: 10.1007/s10207-019-00449-8.
- [80] Hichang Cho, Jae-Shin Lee, and Siyoung Chung. “Optimistic bias about online privacy risks: Testing the moderating effects of perceived controllability and prior experience”. In: *Computers in Human Behavior* 26.5 (2010), pp. 987–995. DOI: 10.1016/j.chb.2010.02.012.
- [81] Haksoo Choi et al. “SensorSafe: A Framework for Privacy-Preserving Management of Personal Sensory Information”. In: *Secure Data Management*. Ed. by Willem Jonker and Milan Petkovi. Vol. 6933. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 85–100. DOI: 10.1007/978-3-642-23556-6_6.
- [82] Arijit Chowdhury et al. “An Improved Fusion Algorithm For Estimating Speed From Smartphone’s Ins/Gps Sensors”. In: *Next Generation Sensors and Systems*. Ed. by Subhas Chandra Mukhopadhyay. Vol. 16. Smart Sensors, Measurement and Instrumentation. Cham: Springer International Publishing, 2016, pp. 235–256. DOI: 10.1007/978-3-319-21671-3_11.
- [83] Maximilian Christ et al. “Time Series Feature Extraction on basis of Scalable Hypothesis tests (tsfresh – A Python package)”. In: *Neurocomputing* 307 (2018), pp. 72–77. DOI: 10.1016/j.neucom.2018.03.067.

- [84] Tzu-Yi Chuang, Nei-Hao Perng, and Jen-Yu Han. "Pavement performance monitoring and anomaly recognition based on crowdsourcing spatiotemporal data". In: *Automation in Construction* 106 (2019), p. 102882. DOI: 10.1016/j.autcon.2019.102882.
- [85] Corona-Warn-App. *Documentation*. 2022. URL: <https://github.com/corona-warn-app/cwa-documentation/tree/88150d82ece4a5f5d1eaa7f7320ef177ee3c6109>.
- [86] Rogier Creemers. "China's Social Credit System: An Evolving Practice of Control". In: *SSRN Electronic Journal* (2018). DOI: 10.2139/ssrn.3175792.
- [87] Emiliano de Cristofaro and Claudio Soriente. "PEPSI—privacy-enhanced participatory sensing infrastructure (Short Paper)". In: *Proceedings of the fourth ACM conference on Wireless network security - WiSec '11*. Ed. by Dieter Gollmann et al. New York, New York, USA: ACM Press, 2011, p. 23. DOI: 10.1145/1998412.1998418.
- [88] Mary J. Culnan and Pamela K. Armstrong. "Information Privacy Concerns, Procedural Fairness, and Impersonal Trust: An Empirical Investigation". In: *Organization Science* 10.1 (1999), pp. 104–115. DOI: 10.1287/orsc.10.1.104.
- [89] Mary J. Culnan and Robert J. Bies. "Consumer Privacy: Balancing Economic and Justice Considerations". In: *Journal of Social Issues* 59.2 (2003), pp. 323–342. DOI: 10.1111/1540-4560.00067.
- [90] Mary J. Culnan and Sandra Milberg. "The Second Exchange: Managing Customer Information in Marketing Relationships". In: *SSRN Electronic Journal* (1998). DOI: 10.2139/ssrn.2621796.
- [91] Jian S. Dai. "Euler–Rodrigues formula variations, quaternion conjugation and intrinsic connections". In: *Mechanism and Machine Theory* 92 (2015), pp. 144–152. DOI: 10.1016/j.mechmachtheory.2015.03.004.
- [92] Hung Dang and Ee-Chien Chang. "PrAd". In: *Proceedings of the 2nd Workshop on Privacy in Geographic Information Collection and Analysis*. Ed. by Grant McKenzie, Krzysztof Janowicz, and Gueorgi Kossinets. New York, NY, USA: Acm, 2015, pp. 1–8. DOI: 10.1145/2830834.2830839.
- [93] Anupam Das, Nikita Borisov, and Matthew Caesar. "Do You Hear What I Hear?" In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. Ed. by Gail-Joon Ahn, Moti Yung, and Ninghui Li. New York, NY, USA: Acm, 2014, pp. 441–452. DOI: 10.1145/2660267.2660325.
- [94] Anupam Das, Nikita Borisov, and Matthew Caesar. "Tracking Mobile Web Users Through Motion Sensors: Attacks and Defenses". In: *Proceedings 2016 Network and Distributed System Security Symposium*. Ed. by Srdjan Capkun. Reston, VA: Internet Society, 2016. DOI: 10.14722/ndss.2016.23390.

- [95] Anupam Das, Nikita Borisov, and Edward Chou. “Every Move You Make: Exploring Practical Issues in Smartphone Motion Sensor Fingerprinting and Countermeasures”. In: *Proceedings on Privacy Enhancing Technologies* 2018.1 (2018), pp. 88–108. DOI: 10.1515/popets-2018-0005.
- [96] Erhan Davarci et al. “Age group detection using smartphone motion sensors”. In: *2017 25th European Signal Processing Conference (EUSIPCO)*. IEEE, 2017, pp. 2201–2205. DOI: 10.23919/eusipco.2017.8081600.
- [97] Ethan W. Dereszynski and Thomas G. Dietterich. “Spatiotemporal Models for Data-Anomaly Detection in Dynamic Environmental Monitoring Campaigns”. In: *ACM Transactions on Sensor Networks* 8.1 (2011), pp. 1–36. DOI: 10.1145/1993042.1993045.
- [98] Android Developers. *Sensors Overview*. 2019. URL: <https://developer.android.com/guide/topics/sensors/sensors%5C%5Foverview>.
- [99] Android Developers. *Declare app permissions*. 2020. URL: <https://developer.android.com/training/permissions/declaring>.
- [100] Android Developers. *Permissions on Android*. 2021. URL: <https://developer.android.com/guide/topics/permissions/overview>.
- [101] Android Developers. *Request app permissions*. 2021. URL: <https://developer.android.com/training/permissions/requesting>.
- [102] Android Developers. *SensorEvent*. 2021. URL: <https://developer.android.com/reference/android/hardware/SensorEvent.html>.
- [103] Google Developers. *Exposure Notifications API*. 2022. URL: <https://developers.google.com/android/exposure-notifications/exposure-notifications-api%5C#architecture>.
- [104] Anind K. Dey. “Understanding and Using Context”. In: *Personal and Ubiquitous Computing* 5.1 (2001), pp. 4–7. DOI: 10.1007/s007790170019.
- [105] Meenu Rani Dey et al. “MagTrack: Detecting Road Surface Condition using Smartphone Sensors and Machine Learning”. In: *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*. IEEE, 2019, pp. 2485–2489. DOI: 10.1109/tencon.2019.8929717.
- [106] Sanorita Dey et al. “AccelPrint: Imperfections of Accelerometers Make Smartphones Trackable”. In: *Proceedings 2014 Network and Distributed System Security Symposium*. Ed. by Lujo Bauer. Reston, VA: Internet Society, 2014. DOI: 10.14722/ndss.2014.23059.
- [107] Michalis Diamantaris et al. “The Seven Deadly Sins of the HTML5 WebAPI”. In: *ACM Transactions on Privacy and Security* 23.4 (2020), pp. 1–31. DOI: 10.1145/3403947.

- [108] Ericson D. Dimaunahan et al. “Road Surface Quality Assessment Using Fast-Fourier Transform”. In: *2020 IEEE 12th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*. IEEE, 2020, pp. 1–5. DOI: 10.1109/hnicem51456.2020.9399996.
- [109] Anuj Dimri et al. “BaroSense”. In: *ACM Transactions on Sensor Networks* 16.1 (2020), pp. 1–24. DOI: 10.1145/3364697.
- [110] Tamara Dinev and Paul Hart. “An Extended Privacy Calculus Model for E-Commerce Transactions”. In: *Information Systems Research* 17.1 (2006), pp. 61–80. DOI: 10.1287/isre.1060.0080.
- [111] Tamara Dinev, Allen R. McConnell, and H. Jeff Smith. “Research Commentary—Informing Privacy Research Through Information Systems, Psychology, and Behavioral Economics: Thinking Outside the “APCO” Box”. In: *Information Systems Research* 26.4 (2015), pp. 639–655. DOI: 10.1287/isre.2015.0600.
- [112] Herbert B. M. Diniz et al. “A Reference Architecture for Mobile Crowdsensing Platforms”. In: *Proceedings of the 18th International Conference on Enterprise Information Systems*. SCITEPRESS - Science, 2016, pp. 600–607. DOI: 10.5220/0005837606000607.
- [113] Viengnam Douangphachanh and Hiroyuki Oneyama. “Estimation of road roughness condition from smartphones under realistic settings”. In: *2013 13th International Conference on ITS Telecommunications (ITST)*. IEEE, 2013, pp. 433–439. DOI: 10.1109/itst.2013.6685585.
- [114] Viengnam Douangphachanh and Hiroyuki Oneyama. “Exploring the Use of Smartphone Accelerometer and Gyroscope to Study on the Estimation of Road Surface Roughness Condition”. In: *Proceedings of the 11th International Conference on Informatics in Control, Automation and Robotics*. SCITEPRESS - Science, 2014, pp. 783–787. DOI: 10.5220/0005117407830787.
- [115] Viengnam Douangphachanh and Hiroyuki Oneyama. “Formulation of a simple model to estimate road surface roughness condition from Android smartphone sensors”. In: *2014 IEEE Ninth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)*. IEEE, 2014, pp. 1–6. DOI: 10.1109/issnip.2014.6827694.
- [116] Akshay Dua, Nirupama Bulusu, and Wu-chang Feng. “Privacy-Preserving Online Mixing of High Integrity Mobile Multi-user Data”. In: *Security and Privacy in Communication Networks*. Ed. by Muttukrishnan Rajarajan et al. Vol. 96. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 470–479. DOI: 10.1007/978-3-642-31909-9_28.

- [117] Wodzisaw Duch. “Filter Methods”. In: *Feature Extraction*. Ed. by Isabelle Guyon et al. Vol. 207. Studies in Fuzziness and Soft Computing. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 89–117. DOI: 10.1007/978-3-540-35488-8_4.
- [118] Hugh F. Durrant-Whyte. “Sensor Models and Multisensor Integration”. In: *The International Journal of Robotics Research* 7.6 (1988), pp. 97–113. DOI: 10.1177/027836498800700608.
- [119] Cynthia Dwork. “Differential Privacy: A Survey of Results”. In: *Theory and Applications of Models of Computation*. Ed. by Manindra Agrawal et al. Vol. 4978. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 1–19. DOI: 10.1007/978-3-540-79228-4_1.
- [120] Erin Egan. *Communicating About Privacy: Towards People-Centered and Accountable Design: CHARTING A WAY FORWARD*. Ed. by Facebook. 2020. URL: <https://about.fb.com/wp-content/uploads/2020/07/Privacy-Transparency-White-Paper.pdf>.
- [121] Miro Enev et al. “Automobile Driver Fingerprinting”. In: *Proceedings on Privacy Enhancing Technologies* 2016.1 (Sept. 2016), pp. 34–50. DOI: 10.1515/popets-2015-0029. URL: <https://doi.org/10.1515%5C%2Fpopets-2015-0029>.
- [122] Jakob Eriksson et al. “The pothole patrol”. In: *Proceeding of the 6th international conference on Mobile systems, applications, and services - MobiSys '08*. Ed. by Dirk Grunwald et al. New York, New York, USA: ACM Press, 2008, p. 29. DOI: 10.1145/1378600.1378605.
- [123] Bilal Esmael et al. “Multivariate Time Series Classification by Combining Trend-Based and Value-Based Approximations”. In: *Computational Science and Its Applications – ICCSA 2012*. Ed. by David Hutchison et al. Vol. 7336. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 392–403. DOI: 10.1007/978-3-642-31128-4_29.
- [124] Adriano Faggiani et al. “Smartphone-based crowdsourcing for network monitoring: Opportunities, challenges, and a case study”. In: *IEEE Communications Magazine* 52.1 (2014), pp. 106–113. DOI: 10.1109/mcom.2014.6710071.
- [125] Mohamed Fazeen et al. “Safe Driving Using Mobile Phones”. In: *IEEE Transactions on Intelligent Transportation Systems* 13.3 (2012), pp. 1462–1468. DOI: 10.1109/tits.2012.2187640.
- [126] Hannes Federrath. “Privacy Enhanced Technologies: Methods – Markets – Misuse”. In: *Trust, Privacy, and Security in Digital Business*. Ed. by David Hutchison et al. Vol. 3592. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 1–9. DOI: 10.1007/11537878_1.
- [127] Adrienne Porter Felt et al. “Android permissions”. In: *Proceedings of the Eighth Symposium on Usable Privacy and Security*. Ed. by Lorrie Faith Cranor. New York, NY: Acm, 2012, p. 1. DOI: 10.1145/2335356.2335360.

- [128] Jianyuan Feng et al. “Hybrid Online Multi-Sensor Error Detection and Functional Redundancy for Artificial Pancreas Control Systems”. In: *IFAC-PapersOnLine* 51.18 (2018), pp. 138–143. DOI: 10.1016/j.ifacol.2018.09.289.
- [129] Simone Fischer-Hbner and Stefan Berthold. “Privacy-Enhancing Technologies”. In: *Computer and Information Security Handbook*. Elsevier, 2017, pp. 759–778. DOI: 10.1016/b978-0-12-803843-7.00053-3.
- [130] Alistair Barrie Forbes and Redouane Boudjemaa. *Parameter estimation methods in data fusion*. Feb. 2014.
- [131] Raisa Z. Freidlin et al. “Measuring Risky Driving Behavior Using an mHealth Smartphone App: Development and Evaluation of gForce”. In: *JMIR mHealth and uHealth* 6.4 (2018), e69. DOI: 10.2196/mhealth.9290.
- [132] Julien Freudiger, Reza Shokri, and Jean-Pierre Hubaux. “Evaluating the Privacy Risk of Location-Based Services”. In: *Financial Cryptography and Data Security*. Ed. by George Danezis. Vol. 7035. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 31–46. DOI: 10.1007/978-3-642-27576-0_3.
- [133] Nathanael C. Fung et al. “Driver identification using vehicle acceleration and deceleration events from naturalistic driving of older drivers”. In: *2017 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*. IEEE, 2017, pp. 33–38. DOI: 10.1109/MeMeA.2017.7985845.
- [134] Stefan Funke and Sabine Storandt. “Path shapes”. In: *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '11*. Ed. by Isabel Cruz and Divyakant Agrawal. New York, New York, USA: ACM Press, 2011, p. 319. DOI: 10.1145/2093973.2094016.
- [135] Bernhard Gahr et al. “Driver Identification via Brake Pedal Signals — A Replication and Advancement of Existing Techniques”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 1415–1420. DOI: 10.1109/itsc.2018.8569510.
- [136] Raghu Ganti, Fan Ye, and Hui Lei. “Mobile crowdsensing: current state and future challenges”. In: *IEEE Communications Magazine* 49.11 (2011), pp. 32–39. DOI: 10.1109/mcom.2011.6069707.
- [137] Xianyi Gao et al. “Elastic pathing”. In: *Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. Ed. by A. J. Brush et al. New York, NY, USA: Acm, 2014, pp. 975–986. DOI: 10.1145/2632048.2632077.
- [138] Xianyi Gao et al. *Transforming Speed Sequences into Road Rays on the Map with Elastic Pathing*. 2017. DOI: 10.48550/ARXIV.1710.06932. URL: <http://arxiv.org/pdf/1710.06932v1>.

- [139] Shahd Mohamed Abdel Gawad, Amr El Mougy, and Menna Ahmed El-Meligy. “Dynamic Mapping of Road Conditions Using Smartphone Sensors and Machine Learning Techniques”. In: *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*. IEEE, 2016, pp. 1–5. DOI: 10.1109/VTCFall.2016.7880972.
- [140] Daniel Genkin, Adi Shamir, and Eran Tromer. “RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis”. In: *Advances in Cryptology – CRYPTO 2014*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 444–461. DOI: 10.1007/978-3-662-44371-2_25.
- [141] Daniel Genkin et al. “Synesthesia: Detecting Screen Content via Remote Acoustic Side Channels”. In: *2019 IEEE Symposium on Security and Privacy*. Piscataway, NJ: IEEE, 2019, pp. 853–869. DOI: 10.1109/sp.2019.00074.
- [142] Nina Gerber, Paul Gerber, and Melanie Volkamer. “Explaining the privacy paradox: A systematic review of literature investigating privacy attitude and behavior”. In: *Computers & Security* 77 (2018), pp. 226–261. DOI: 10.1016/j.cose.2018.04.002.
- [143] Amrita Ghosal and Mauro Conti. “Security issues and challenges in V2X: A Survey”. In: *Computer Networks* 169 (2020), p. 107093. DOI: 10.1016/j.comnet.2019.107093.
- [144] Avik Ghose et al. “An enhanced automated system for evaluating harsh driving using smartphone sensors”. In: *Proceedings of the 17th International Conference on Distributed Computing and Networking*. New York, NY, USA: Acm, 2016, pp. 1–6. DOI: 10.1145/2833312.2849555.
- [145] Dibyajyoti Ghosh et al. “Privacy Control in Smart Phones Using Semantically Rich Reasoning and Context Modeling”. In: *2012 IEEE Symposium on Security and Privacy Workshops*. IEEE, 2012, pp. 82–85. DOI: 10.1109/spw.2012.27.
- [146] Rashi Glazer. “Marketing in an Information-Intensive Environment: Strategic Implications of Knowledge as an Asset”. In: *Journal of Marketing* 55.4 (1991), p. 1. DOI: 10.2307/1251953.
- [147] Michaela Götz, Suman Nath, and Johannes Gehrke. “MaskIt”. In: *Proceedings of the 2012 international conference on Management of Data - SIGMOD '12*. Ed. by K. Selçuk Candan et al. New York, New York, USA: ACM Press, 2012, p. 289. DOI: 10.1145/2213836.2213870.
- [148] Michelle Greenwood and Harry J. van Buren III. “Trust and Stakeholder Theory: Trustworthiness in the Organisation–Stakeholder Relationship”. In: *Journal of Business Ethics* 95.3 (2010), pp. 425–438. DOI: 10.1007/s10551-010-0414-4.
- [149] Joe Grengs, Xiaoguang Wang, and Lidia Kostyniuk. “Using GPS Data to Understand Driving Behavior”. In: *Journal of Urban Technology* 15.2 (2008), pp. 33–53. DOI: 10.1080/10630730802401942.

- [150] Patrizia Grifoni, Arianna D’Ulizia, and Fernando Ferri. “Context-Awareness in Location Based Services in the Big Data Era”. In: *Mobile Big Data*. Ed. by Georgios Skourletopoulos et al. Vol. 10. Lecture Notes on Data Engineering and Communications Technologies. Cham: Springer International Publishing, 2018, pp. 85–127. DOI: 10.1007/978-3-319-67925-9_5.
- [151] Einat Grimberg, Assaf Botzer, and Oren Musicant. “Smartphones vs. in-vehicle data acquisition systems as tools for naturalistic driving studies: A comparative review”. In: *Safety Science* 131 (2020), p. 104917. DOI: 10.1016/j.ssci.2020.104917.
- [152] Isaac Griswold-Steiner, Zachary LeFevre, and Abdul Serwadda. “Smartphone speech privacy concerns from side-channel attacks on facial biomechanics”. In: *Computers & Security* 100 (2021), p. 102110. DOI: 10.1016/j.cose.2020.102110.
- [153] Monica Grosso et al. “What Information Do Shoppers Share? The Effect of Personnel-, Retailer-, and Country-Trust on Willingness to Share Information”. In: *Journal of Retailing* 96.4 (2020), pp. 524–547. DOI: 10.1016/j.jretai.2020.08.002.
- [154] Marco Gruteser and Dirk Grunwald. “Anonymous Usage of Location-Based Services Through Spatial and Temporal Cloaking”. In: *Proceedings of the 1st international conference on Mobile systems, applications and services - MobiSys ’03*. Ed. by Dan Siewiorek, Mary Baker, and Robert T. Morris. New York, New York, USA: ACM Press, 2003, pp. 31–42. DOI: 10.1145/1066116.1189037.
- [155] Yanlei Gu, Qianlong Wang, and Shunsuke Kamijo. “Intelligent Driving Data Recorder in Smartphone Using Deep Neural Network-Based Speedometer and Scene Understanding”. In: *IEEE Sensors Journal* 19.1 (2019), pp. 287–296. DOI: 10.1109/jsen.2018.2874665.
- [156] Alec Guertin and Vadim Kotov. *PHA Family Highlights: Bread (and Friends)*. Ed. by Android Security & Privacy Team. 2020. URL: <https://security.googleblog.com/2020/01/pha-family-highlights-bread-and-friends.html>.
- [157] Bin Guo et al. “From participatory sensing to Mobile Crowd Sensing”. In: *2014 IEEE International Conference on Pervasive Computing and Communication workshops (PerCom workshops 2014)*. Piscataway, NJ: IEEE, 2014, pp. 593–598. DOI: 10.1109/PerComW.2014.6815273.
- [158] Abhishek Gupta et al. “Road grade estimation using crowd-sourced smartphone data”. In: *Proceedings - 2020 19th ACM/IEEE International Conference on Information Processing in Sensor Networks, IPSN 2020*. Institute of Electrical and Electronics Engineers Inc, 2020, pp. 313–324.
- [159] Haritabh Gupta et al. “Deciphering Text from Touchscreen Key Taps”. In: *Data and Applications Security and Privacy XXX*. Ed. by Silvio Ranise and Vipin Swarup. Vol. 9766. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2016, pp. 3–18. DOI: 10.1007/978-3-319-41483-6_1.

- [160] Seda Gürses, Carmela Troncoso, and Claudia Diaz. “Engineering Privacy by Design Reloaded”. In: *Amsterdam Privacy Conf* (2015).
- [161] David L. Hall and Sonya Anne Hall McMullen. *Mathematical techniques in multisensor data fusion*. 2. ed. Boston: Artech House, 2004.
- [162] David Hallac et al. “Driver identification using automobile sensor data from a single turn”. In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, Nov. 2016, pp. 953–958. DOI: 10.1109/itsc.2016.7795670. URL: <https://doi.org/10.1109%5C%2Fitsc.2016.7795670>.
- [163] Jun Han et al. “ACComplice: Location inference using accelerometers on smartphones”. In: *2012 Fourth International Conference on Communication Systems and Networks (COMSNETS 2012)*. IEEE, 2012, pp. 1–9. DOI: 10.1109/comsnets.2012.6151305.
- [164] Meng Han et al. “Near-Complete Privacy Protection: Cognitive Optimal Strategy in Location-Based Services”. In: *Procedia Computer Science* 129 (2018), pp. 298–304. DOI: 10.1016/j.procs.2018.03.079.
- [165] Weili Han et al. “senDroid: Auditing Sensor Access in Android System-Wide”. In: *IEEE Transactions on Dependable and Secure Computing* 17.2 (2020), pp. 407–421. DOI: 10.1109/tdsc.2017.2768536.
- [166] P. M. Harikrishnan and Varun P. Gopi. “Vehicle Vibration Signal Processing for Road Surface Monitoring”. In: *IEEE Sensors Journal* 17.16 (2017), pp. 5192–5197. DOI: 10.1109/jssen.2017.2719865.
- [167] Woodrow Hartzog. *Privacy’s blueprint: The battle to control the design of new technologies*. Cambridge, Massachusetts and London: Harvard University Press, 2018.
- [168] Ulrich Hedtstück. *Complex Event Processing: Verarbeitung von Ereignismustern in Datenströmen*. eXamen.press. Berlin: Springer Vieweg, 2017.
- [169] Markus Herb et al. “Crowd-sourced Semantic Edge Mapping for Autonomous Vehicles”. In: *IEEE International Conference on Intelligent Robots and Systems*. Institute of Electrical and Electronics Engineers Inc, 2019, pp. 7047–7053.
- [170] Johannes Heurix et al. “A taxonomy for privacy enhancing technologies”. In: *Computers & Security* 53 (2015), pp. 1–17. DOI: 10.1016/j.cose.2015.05.002.
- [171] Alan Hevner and Samir Chatterjee. *Design Research in Information Systems*. Vol. 22. Boston, MA: Springer US, 2010. DOI: 10.1007/978-1-4419-5653-8.
- [172] Bo-Jhang Ho et al. “From Pressure to Path: Barometer-based Vehicle Tracking”. In: *BuildSys’15 : proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Buildings : November 4-5, 2015, Seoul, South Korea. ACM Conference on Embedded Systems for Energy-Efficient Buildings (2nd : 2015... 2015* (2015), pp. 65–74. DOI: 10.1145/2821650.2821665.

- [173] Duncan Hodges and Oliver Buckley. “Reconstructing What You Said: Text Inference Using Smartphone Motion”. In: *IEEE Transactions on Mobile Computing* 18.4 (2019), pp. 947–959. DOI: 10.1109/tmc.2018.2850313.
- [174] Jaap-Henk Hoepman. “Privacy Design Strategies”. In: *ICT Systems Security and Privacy Protection*. Ed. by Nora Cuppens-Boulahia et al. Vol. 428. IFIP Advances in Information and Communication Technology. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 446–459. DOI: 10.1007/978-3-642-55415-5_38.
- [175] Geert Hofstede. *Culture’s consequences: Comparing values behaviors institutions and organizations across nations*. 2. ed. [Nachdr.] Thousand Oaks [u.a.]: Sage Publ, 2008.
- [176] Baik Hoh et al. “Enhancing Security and Privacy in Traffic-Monitoring Systems”. In: *IEEE Pervasive Computing* 5.4 (2006), pp. 38–46. DOI: 10.1109/mpvr.2006.69.
- [177] Baik Hoh et al. “Preserving privacy in gps traces via uncertainty-aware path cloaking”. In: *Proceedings of the 14th ACM conference on Computer and communications security - CCS ’07*. Ed. by Peng Ning, Sabrina de Di Capitani Vimercati, and Paul Syverson. New York, New York, USA: ACM Press, 2007, p. 161. DOI: 10.1145/1315245.1315266.
- [178] Baik Hoh et al. “Virtual trip lines for distributed privacy-preserving traffic monitoring”. In: *Proceeding of the 6th international conference on Mobile systems, applications, and services - MobiSys ’08*. Ed. by Dirk Grunwald et al. New York, New York, USA: ACM Press, 2008, p. 15. DOI: 10.1145/1378600.1378604.
- [179] Jeff Howe. *Crowdsourcing: A Definition*. 2006. URL: <https://crowdsourcing.typepad.com/cs/2006/06/crowdsourcing%5C%5Fa.html>.
- [180] Jeff Howe. “The Rise of Crowdsourcing”. In: *Wired* 2006.14 (2006). URL: <http://www.wired.com/wired/archive/14.06/crowds.html>.
- [181] Jingyu Hua, Zhenyu Shen, and Sheng Zhong. “We Can Track You if You Take the Metro: Tracking Metro Riders Using Accelerometers on Smartphones”. In: *IEEE Transactions on Information Forensics and Security* 12.2 (2017), pp. 286–297. DOI: 10.1109/tifs.2016.2611489.
- [182] Xiang-hua Huang. “Sensor Fault Diagnosis and Reconstruction of Engine Control System Based on Autoassociative Neural Network”. In: *Chinese Journal of Aeronautics* 17.1 (2004), pp. 23–27. DOI: 10.1016/s1000-9361(11)60198-2.
- [183] Lars Huning, Jan Bauer, and Nils Aschenbruck. “A Privacy Preserving Mobile Crowdsensing Architecture for a Smart Farming Application”. In: *Proceedings of the First ACM Workshop on Mobile Crowdsensing Systems and Applications*. Ed. by Rasit Eskicioglu. New York, NY, USA: Acm, 2017, pp. 62–67. DOI: 10.1145/3139243.3139250.

- [184] Sinia Husnjak et al. “Telematics System in Usage Based Motor Insurance”. In: *Procedia Engineering* 100 (2015), pp. 816–825. DOI: 10.1016/j.proeng.2015.01.436. URL: <https://doi.org/10.1016%5C%2Fj.proeng.2015.01.436>.
- [185] Muzammil Hussain et al. “The rise of keyloggers on smartphones: A survey and insight into motion-based tap inference attacks”. In: *Pervasive and Mobile Computing* 25 (2016), pp. 1–25. DOI: 10.1016/j.pmcj.2015.12.001.
- [186] Vinay Ijure and Ronald Williams. “Taxonomies of attacks and vulnerabilities in computer systems”. In: *IEEE Communications Surveys & Tutorials* 10.1 (2008), pp. 6–19. DOI: 10.1109/comst.2008.4483667.
- [187] Honeywell International Inc. *Honeywell AN-203 - Compass Heading Using Magnetometers: Technical Report*. 2004. URL: <https://cdn-shop.adafruit.com/datasheets/AN203%5C%5FCompass%5C%5FHeading%5C%5FUsing%5C%5FMagnetometers.pdf>.
- [188] Ibrahim M. Al-Jabri, Mustafa I. Eid, and Amer Abed. “The willingness to disclose personal information”. In: *Information & Computer Security* 28.2 (2019), pp. 161–181. DOI: 10.1108/ics-01-2018-0012.
- [189] Sasan Jafarnejad, German Castignani, and Thomas Engel. “Towards a real-time driver identification mechanism based on driving sensing data”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, Oct. 2017, pp. 1–7. DOI: 10.1109/itsc.2017.8317716. URL: <https://doi.org/10.1109%5C%2Fitsc.2017.8317716>.
- [190] Sasan Jafarnejad, German Castignani, and Thomas Engel. “Revisiting Gaussian Mixture Models for Driver Identification”. In: *2018 IEEE International Conference on Vehicular Electronics and Safety (ICVES)*. IEEE, 2018, pp. 1–7. DOI: 10.1109/icves.2018.8519588.
- [191] Tun-Min Jai and Nancy J. King. “Privacy versus reward: Do loyalty programs increase consumers’ willingness to share personal information with third-party advertisers and data brokers?” In: *Journal of Retailing and Consumer Services* 28 (2016), pp. 296–303. DOI: 10.1016/j.jretconser.2015.01.005.
- [192] Abdul Rehman Javed et al. “AlphaLogger: detecting motion-based side-channel attack using smartphone keystrokes”. In: *Journal of Ambient Intelligence and Humanized Computing* (2020). DOI: 10.1007/s12652-020-01770-0.
- [193] Daun Jeong et al. “Real-time Driver Identification using Vehicular Big Data and Deep Learning”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 123–130. DOI: 10.1109/itsc.2018.8569452.
- [194] Théo Jourdan, Antoine Boutet, and Carole Frindel. “Toward privacy in IoT mobile devices for activity recognition”. In: *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. Ed. by Henning Schulzrinne and Pan Li. New York, NY, USA: Acm, 2018, pp. 155–165. DOI: 10.1145/3286978.3287009.

- [195] Aya El-Kady et al. “Road Surface Quality Detection using Smartphone Sensors: Egyptian Roads Case Study”. In: *2019 Ninth International Conference on Intelligent Computing and Information Systems (ICICIS)*. IEEE, 2019, pp. 202–207. DOI: 10.1109/icicis46948.2019.9014721.
- [196] Daniel Kahneman, Jack L. Knetsch, and Richard H. Thaler. “Anomalies: The Endowment Effect, Loss Aversion, and Status Quo Bias”. In: *Journal of Economic Perspectives* 5.1 (1991), pp. 193–206. DOI: 10.1257/jep.5.1.193.
- [197] Faria Kalim, Jaehoon Paul Jeong, and Muhammad U. Ilyas. “CRATER: A Crowd Sensing Application to Estimate Road Conditions”. In: *IEEE Access* 4 (2016), pp. 8317–8326. DOI: 10.1109/access.2016.2607719.
- [198] Stratis Kanarachos, Stavros-Richard G. Christopoulos, and Alexander Chroneos. “Smartphones as an integrated platform for monitoring driver behaviour: The role of sensor fusion and connectivity”. In: *Transportation Research Part C: Emerging Technologies* 95 (2018), pp. 867–882. DOI: 10.1016/j.trc.2018.03.023.
- [199] Lei Kang and Suman Banerjee. “Practical driving analytics with smartphone sensors”. In: *2017 IEEE Vehicular Networking Conference (VNC)*. IEEE, 2017, pp. 303–310. DOI: 10.1109/vnc.2017.8275595.
- [200] Salil S. Kanhere. “Participatory Sensing: Crowdsourcing Data from Mobile Smartphones in Urban Spaces”. In: *2011 IEEE 12th International Conference on Mobile Data Management*. IEEE, 2011, pp. 3–6. DOI: 10.1109/mdm.2011.16.
- [201] Charles F. F. Karney. “Algorithms for geodesics”. In: *Journal of Geodesy* 87.1 (2013), pp. 43–55. DOI: 10.1007/s00190-012-0578-z.
- [202] Sabrina Karwatzki et al. “Beyond the Personalization–Privacy Paradox: Privacy Valuation, Transparency Features, and Service Personalization”. In: *Journal of Management Information Systems* 34.2 (2017), pp. 369–400. DOI: 10.1080/07421222.2017.1334467.
- [203] Kotaro Kataoka et al. “A Smartphone-Based Probe Data Platform for Road Management and Safety in Developing Countries”. In: *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*. IEEE, 2018, pp. 612–615. DOI: 10.1109/icdmw.2018.00095.
- [204] Eamonn Keogh et al. “Segmenting time series: A survey and novel approach”. In: *Data Mining in Time Series Databases*. Vol. Volume 57. Series in Machine Perception and Artificial Intelligence. World Scientific, 2004, pp. 1–21. DOI: 10.1142/9789812565402_0001.
- [205] Chaker Abdelaziz Kerrache et al. “Trust Management for Vehicular Networks: An Adversary-Oriented Overview”. In: *IEEE Access* 4 (2016), pp. 9293–9307. DOI: 10.1109/access.2016.2645452.
- [206] Bahador Khaleghi et al. “Multisensor data fusion: Antecedents and directions”. In: *2009 3rd International Conference on Signals, Circuits and Systems (SCS)*. IEEE, 2009, pp. 1–6. DOI: 10.1109/icscs.2009.5412296.

- [207] Byoungsoo Kim and Daekil Kim. “Understanding the Key Antecedents of Users’ Disclosing Behaviors on Social Networking Sites: The Privacy Paradox”. In: *Sustainability* 12.12 (2020), p. 5163. DOI: 10.3390/su12125163.
- [208] Hyosu Kim et al. “TapSnoop: Inferring Tapstrokes from Listening to Tap Sound on Mobile Devices”. In: *Proceedings of the 14th Annual International Conference on Mobile Systems, Applications, and Services Companion - MobiSys '16 Companion*. Ed. by Rajesh Balan et al. New York, New York, USA: ACM Press, 2016, p. 137. DOI: 10.1145/2938559.2938595.
- [209] Barbara Kitchenham and Stuart Charters. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Keele University, University of Durham Keele, and Durham, UK, 2007.
- [210] Lawrence A. Klein. *Sensor and data fusion concepts and applications*. Bellingham, Wash., 1999.
- [211] Bart P. Knijnenburg, Alfred Kobsa, and Hongxia Jin. “Dimensionality of information disclosure behavior”. In: *International Journal of Human-Computer Studies* 71.12 (2013), pp. 1144–1162. DOI: 10.1016/j.ijhcs.2013.06.003.
- [212] Spyros Kokolakis. “Privacy attitudes and privacy behaviour: A review of current research on the privacy paradox phenomenon”. In: *Computers & Security* 64 (2017), pp. 122–134. DOI: 10.1016/j.cose.2015.07.002.
- [213] Maheshwari Kotha et al. “PotSense”. In: *Proceedings of the 1st ACM Workshop on Autonomous and Intelligent Mobile Systems*. New York, NY, USA: Acm, 2020, pp. 1–6. DOI: 10.1145/3377283.3377286.
- [214] Hanna Krasnova, Natasha F. Veltri, and Oliver Günther. “Self-disclosure and Privacy Calculus on Social Networking Sites: The Role of Culture”. In: *Business & Information Systems Engineering* 4.3 (2012), pp. 127–135. DOI: 10.1007/s12599-012-0216-6.
- [215] Frauke Kreuter et al. “Collecting Survey and Smartphone Sensor Data With an App: Opportunities and Challenges Around Privacy and Informed Consent”. In: *Social Science Computer Review* 38.5 (2020), pp. 533–549. DOI: 10.1177/0894439318816389.
- [216] Jacob Leon Kröger, Otto Hans-Martin Lutz, and Stefan Ullrich. “The myth of individual control: Mapping the limitations of privacy self-management”. In: *SSRN Electronic Journal* (2021). DOI: 10.2139/ssrn.3881776.
- [217] John Krumm. “Inference Attacks on Location Tracks”. In: *Pervasive Computing*. Ed. by Anthony LaMarca, Marc Langheinrich, and Khai N. Truong. Vol. 4480. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 127–143. DOI: 10.1007/978-3-540-72037-9_8.

- [218] Brian Krupp, Dan Jesenseky, and Amanda Szampias. “SPEProxy: Enforcing fine grained security and privacy controls on unmodified mobile devices”. In: *2017 IEEE 8th Annual Ubiquitous Computing, Electronics and Mobile Communication Conference (UEMCON)*. IEEE, 2017, pp. 520–526. DOI: 10.1109/uemcon.2017.8248985.
- [219] Tim Kuhlmann, Pablo Garaizar, and Ulf-Dietrich Reips. “Smartphone sensor accuracy varies from device to device in mobile research: The case of spatial orientation”. In: *Behavior research methods* 53.1 (2021), pp. 22–33. DOI: 10.3758/s13428-020-01404-5.
- [220] Jack B. Kuipers. *Quaternions and rotation sequences: A primer with applications to orbits, aerospace, and virtual reality*. Princeton, N.J.: Princeton University Press, 1999. URL: <http://www.loc.gov/catdir/description/prin021/98035389.html>.
- [221] Saurabh Kumar and Sandeep Kumar Shukla. “The State of Android Security”. In: *Cyber Security in India*. Ed. by Sandeep Kumar Shukla and Manindra Agrawal. IITK Directions. Singapore: Springer Singapore, 2020, pp. 17–22. DOI: 10.1007/978-981-15-1675-7_2.
- [222] Thusithanjana Kavinda Kumara Thilakarathna, Hasith E. Perera, and H. H. E. Jayaweera. “A mobile application for crowdsourced road condition monitoring”. In: *2019 4th International Conference on Information Technology Research (ICITR)*. IEEE, 2019, pp. 1–4. DOI: 10.1109/icitr49409.2019.9407782.
- [223] Byung Il Kwak, JiYoung Woo, and Huy Kang Kim. “Know your master: Driver profiling-based anti-theft method”. In: *2016 14th Annual Conference on Privacy, Security and Trust (PST)*. IEEE, 2016, pp. 211–218. DOI: 10.1109/pst.2016.7906929.
- [224] Charalambos Kyriakou, Symeon E. Christodoulou, and Loukas Dimitriou. “Do Vehicles Sense, Detect and Locate Speed Bumps?” In: *Transportation Research Procedia* 52 (2021), pp. 203–210. DOI: 10.1016/j.trpro.2021.01.023.
- [225] Paul F. Langer. “Lessons from China - The Formation of a Social Credit System: Profiling, Reputation Scoring, Social Engineering”. In: *The 21st Annual International Conference on Digital Government Research*. Ed. by Seok-Jin Eom and Jooho Lee. New York, NY, USA: Acm, 2020, pp. 164–174. DOI: 10.1145/3396956.3396962.
- [226] Robert S. Laufer and Maxine Wolfe. “Privacy as a Concept and a Social Issue: A Multidimensional Developmental Theory”. In: *Journal of Social Issues* 33.3 (1977), pp. 22–42. DOI: 10.1111/j.1540-4560.1977.tb01880.x.
- [227] Puttipong Leakkaw and Sooksan Panichpapiboon. “Speed estimation through mobile sensing”. In: *TENCON 2014 - 2014 IEEE Region 10 Conference*. IEEE, 2014, pp. 1–5. DOI: 10.1109/tencon.2014.7022319.

- [228] Pedro Giovanni Leon et al. “What matters to users?” In: *Proceedings of the Ninth Symposium on Usable Privacy and Security - SOUPS '13*. Ed. by Lorrie Faith Cranor, Lujio Bauer, and Konstantin Beznosov. New York, New York, USA: ACM Press, 2013, p. 1. DOI: 10.1145/2501604.2501611.
- [229] Han Li, Rathindra Sarathy, and Heng Xu. “Understanding Situational Online Information Disclosure as a Privacy Calculus”. In: *Journal of Computer Information Systems* 51.1 (2010), pp. 62–71. DOI: 10.1080/08874417.2010.11645450.
- [230] Xiao Li and Daniel W. Goldberg. “Toward a mobile crowdsensing system for road surface assessment”. In: *Computers, Environment and Urban Systems* 69 (2018), pp. 51–62. DOI: 10.1016/j.compenvurbsys.2017.12.005.
- [231] Zi Li et al. “Location Privacy Violation via GPS-agnostic Smart Phone Car Tracking”. In: *IEEE Transactions on Vehicular Technology* (2018), p. 1. DOI: 10.1109/tvt.2018.2800123.
- [232] Liang Cai and Hao Chen. “TouchLogger: Inferring Keystrokes on Touch Screen from Smartphone Motion”. In: 2011. URL: <https://www.usenix.org/conference/hotsec11/touchlogger-inferring-keystrokes-touch-screen-smartphone-motion>.
- [233] Lucas Cedro Lima et al. “Using Crowdsourcing Techniques and Mobile Devices for Asphaltic Pavement Quality Recognition”. In: *2016 VI Brazilian Symposium on Computing Systems Engineering (SBESC)*. IEEE, 2016, pp. 144–149. DOI: 10.1109/sbesc.2016.029.
- [234] Martin Lindfors et al. “Vehicle speed tracking using chassis vibrations”. In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2016, pp. 214–219. DOI: 10.1109/ivs.2016.7535388.
- [235] Wei Liu et al. “Real-Time Traffic Light Recognition Based on Smartphone Platforms”. In: *IEEE Transactions on Circuits and Systems for Video Technology* 27.5 (2017), pp. 1118–1131.
- [236] Silas C. Lobo et al. *INTAS – The Ingolstadt Traffic Scenario for SUMO*. 2020. DOI: 10.48550/ARXIV.2011.11995. URL: <https://arxiv.org/pdf/2011.11995>.
- [237] David C. Luckham. *The power of events: An introduction to complex event processing in distributed enterprise systems*. 3. print. Boston [u.a.]: Addison-Wesley, 2005.
- [238] Chengwen Luo et al. “Predictable Privacy-Preserving Mobile Crowd Sensing: A Tale of Two Roles”. In: *IEEE/ACM Transactions on Networking* 27.1 (2019), pp. 361–374. DOI: 10.1109/tnet.2019.2890860.
- [239] Chu Luo et al. “A data hiding approach for sensitive smartphone data”. In: *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing*. Ed. by Paul Lukowicz et al. New York, NY, USA: Acm, 2016, pp. 557–568. DOI: 10.1145/2971648.2971686.

- [240] Lingjuan Lyu et al. “Privacy-Preserving Collaborative Deep Learning with Application to Human Activity Recognition”. In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. Ed. by Ee-Peng Lim et al. New York, NY, USA: Acm, 2017, pp. 1219–1228. DOI: 10.1145/3132847.3132990.
- [241] Huadong Ma, Dong Zhao, and Peiyan Yuan. “Opportunities in mobile crowd sensing”. In: *IEEE Communications Magazine* 52.8 (2014), pp. 29–35. DOI: 10.1109/mcom.2014.6871666.
- [242] Mohammad Malekzadeh et al. “Protecting Sensory Data against Sensitive Inferences”. In: *Proceedings of the 1st Workshop on Privacy by Design in Distributed Systems*. Ed. by Francisco Maia, Hugues Mercier, and Andrey Brito. New York, NY, USA: Acm, 2018, pp. 1–6. DOI: 10.1145/3195258.3195260.
- [243] Miguel Malheiros, Sören Preibusch, and M. Angela Sasse. ““Fairly Truthful”: The Impact of Perceived Effort, Fairness, Relevance, and Sensitivity on Personal Data Disclosure”. In: *Trust and Trustworthy Computing*. Ed. by David Hutchison et al. Vol. 7904. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 250–266. DOI: 10.1007/978-3-642-38908-5_19.
- [244] Stratos Mansalis et al. “An evaluation of data stream clustering algorithms”. In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 11.4 (2018), pp. 167–187. DOI: 10.1002/sam.11380.
- [245] M. Mansouri et al. “Midpoint-radii principal component analysis -based EWMA and application to air quality monitoring network”. In: *Chemometrics and Intelligent Laboratory Systems* 175 (2018), pp. 55–64. DOI: 10.1016/j.chemolab.2018.01.016.
- [246] Francesco Marcantoni et al. “A Large-scale Study on the Risks of the HTML5 WebAPI for Mobile Sensor-based Attacks”. In: *The World Wide Web Conference on - WWW '19*. Ed. by Ling Liu and Ryen White. New York, New York, USA: ACM Press, 2019, pp. 3063–3071. DOI: 10.1145/3308558.3313539.
- [247] Letizia Marchegiani and Ingmar Posner. “Long-Term Driving Behaviour Modelling for Driver Identification”. In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, Nov. 2018, pp. 913–919. DOI: 10.1109/itsc.2018.8569610. URL: <https://doi.org/10.1109%5C%2Fitsc.2018.8569610>.
- [248] Clara Marina Martinez et al. “Driving Style Recognition for Intelligent Vehicle Control and Advanced Driver Assistance: A Survey”. In: *IEEE Transactions on Intelligent Transportation Systems* 19.3 (Mar. 2018), pp. 666–676. DOI: 10.1109/tits.2017.2706978. URL: <https://doi.org/10.1109%5C%2Ftits.2017.2706978>.
- [249] Ereni Markos, George R. Milne, and James W. Peltier. “Information Sensitivity and Willingness to Provide Continua: A Comparative Privacy Study of the United States and Brazil”. In: *Journal of Public Policy & Marketing* 36.1 (2017), pp. 79–96. DOI: 10.1509/jppm.15.159.

- [250] Ian D. Markwood and Yao Liu. “Vehicle Self-Surveillance”. In: *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*. Ed. by Xiaofeng Chen, XiaoFeng Wang, and Xinyi Huang. New York, NY, USA: Acm, May 2016, pp. 425–436. DOI: 10.1145/2897845.2897917. URL: <https://doi.org/10.1145%5C%2F2897845.2897917>.
- [251] Philip Marquardt et al. “(sp)iPhone”. In: *Proceedings of the 18th ACM conference on Computer and communications security - CCS '11*. Ed. by Yan Chen, George Danezis, and Vitaly Shmatikov. New York, New York, USA: ACM Press, 2011, p. 551. DOI: 10.1145/2046707.2046771.
- [252] Mart'n Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. URL: <https://www.tensorflow.org/>.
- [253] Kirsten E. Martin and Helen Nissenbaum. “Measuring Privacy: Using Context to Expose Confounding Variables”. In: *SSRN Electronic Journal* (2015). DOI: 10.2139/ssrn.2709584.
- [254] M. V. Martinez, J. Echanobe, and I. Del Campo. “Driver identification and impostor detection based on driving behavior signals”. In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, Nov. 2016, pp. 372–378. DOI: 10.1109/itsc.2016.7795582. URL: <https://doi.org/10.1109%5C%2Fitsc.2016.7795582>.
- [255] Alice Marwick and Eszter Hargittai. “Nothing to hide, nothing to lose? Incentives and disincentives to sharing information with institutions online”. In: *Information, Communication & Society* 22.12 (2019), pp. 1697–1713. DOI: 10.1080/1369118x.2018.1450432.
- [256] Johannes Masino et al. “Learning from the crowd: Road infrastructure monitoring system”. In: *Journal of Traffic and Transportation Engineering (English Edition)* 4.5 (2017), pp. 451–463.
- [257] Nikolay Matyunin, Yujue Wang, and Stefan Katzenbeisser. “Vibrational Covert Channels using Low-Frequency Acoustic Signals”. In: *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*. Ed. by Rémi Cogramne et al. New York, NY, USA: Acm, 2019, pp. 31–36. DOI: 10.1145/3335203.3335712.
- [258] Nikolay Matyunin et al. “Tracking Private Browsing Sessions using CPU-based Covert Channels”. In: *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. Ed. by Panos Papadimitratos, Kevin Butler, and Christina Pöpper. New York, NY, USA: Acm, 2018, pp. 63–74. DOI: 10.1145/3212480.3212489.
- [259] Nikolay Matyunin et al. “MagneticSpy”. In: *Proceedings of the 18th ACM Workshop on Privacy in the Electronic Society - WPES'19*. Ed. by Lorenzo Cavallaro, Johannes Kinder, and Josep Domingo-Ferrer. New York, New York, USA: ACM Press, 2019, pp. 135–149. DOI: 10.1145/3338498.3358650.

- [260] William McGeeveran. “The Law of Friction”. In: *University of Chicago Legal Forum* (2013), pp. 15–68.
- [261] Artis Mednis et al. “Real time pothole detection using Android smartphones with accelerometers”. In: *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*. IEEE, 2011, pp. 1–6. DOI: 10.1109/dcoss.2011.5982206.
- [262] Maryam Mehrnezhad et al. “TouchSignatures: Identification of user touch actions and PINs based on mobile sensor data via JavaScript”. In: *Journal of Information Security and Applications* 26 (2016), pp. 23–38. DOI: 10.1016/j.jisa.2015.11.007.
- [263] Alexander Mense et al. “Open Source Based Privacy-Proxy to Restrain Connectivity of Mobile Apps”. In: *Proceedings of the 14th International Conference on Advances in Mobile Computing and Multi Media*. Ed. by Bessam Abdulrazak et al. New York, NY, USA: Acm, 2016, pp. 284–287. DOI: 10.1145/3007120.3007163.
- [264] Chenglin Miao et al. “Privacy-Preserving Truth Discovery in Crowd Sensing Systems”. In: *ACM Transactions on Sensor Networks* 15.1 (2019), pp. 1–32. DOI: 10.1145/3277505.
- [265] Emiliano Miluzzo et al. “Tapprints”. In: *Proceedings of the 10th international conference on Mobile systems, applications, and services - MobiSys '12*. Ed. by Nigel Davies, Srinivasan Seshan, and Lin Zhong. New York, New York, USA: ACM Press, 2012, p. 323. DOI: 10.1145/2307636.2307666.
- [266] Saeed Mirzamohammadi and Ardalan Amiri Sani. “Viola: Trustworthy Sensor Notifications for Enhanced Privacy on Mobile Systems”. In: *IEEE Transactions on Mobile Computing* 17.11 (2018), pp. 2689–2702. DOI: 10.1109/tmc.2018.2812706.
- [267] Chiyomi Miyajima et al. “Driver Modeling Based on Driving Behavior and Its Evaluation in Driver Identification”. In: *Proceedings of the IEEE* 95.2 (2007), pp. 427–437. DOI: 10.1109/jproc.2006.888405.
- [268] Prashanth Mohan, Venkata N. Padmanabhan, and Ramachandran Ramjee. “Nericell”. In: *Proceedings of the 6th ACM conference on Embedded network sensor systems - SenSys '08*. Ed. by Tarek Abdelzaher, Margaret Martonosi, and Adam Wolisz. New York, New York, USA: ACM Press, 2008, p. 323. DOI: 10.1145/1460412.1460444.
- [269] Jean-Philippe Monteuis et al. “Attacker model for connected and automated vehicles”. In: *ACM Computer Science in Car Symposium*. 2018. DOI: 10.1145/3273946.3273951.

- [270] Ladan Mozaffari, Ahmad Mozaffari, and Nasser L. Azad. “Vehicle speed prediction via a sliding-window time series analysis and an evolutionary least learning machine: A case study on San Francisco urban roads”. In: *Engineering Science and Technology, an International Journal* 18.2 (2015), pp. 150–162. DOI: 10.1016/j.jestch.2014.11.002.
- [271] Meinard Müller. *Information retrieval for music and motion*. Berlin, Heidelberg: Springer-Verlag BerlinHeidelberg, 2007. DOI: 10.1007/978-3-540-74048-3.
- [272] Nithin Murali and Kumar Appaiah. “Keyboard Side Channel Attacks on Smartphones Using Sensor Fusion”. In: *2018 IEEE Global Communications Conference (GLOBECOM)*. Piscataway, NJ: IEEE, 2018, pp. 206–212. DOI: 10.1109/glocom.2018.8647336.
- [273] Alexios Mylonas, Anastasia Kastania, and Dimitris Gritzalis. “Delegate the smartphone user? Security awareness in smartphone platforms”. In: *Computers & Security* 34 (2013), pp. 47–66. DOI: 10.1016/j.cose.2012.11.004.
- [274] Sara N-Marandi. *Android Developers Blog: What’s new in Android Privacy*. Android Developers Blog, 2021. URL: <https://android-developers.googleblog.com/2021/05/android-security-and-privacy-recap.html>.
- [275] Sashank Narain, Amirali Sanatinia, and Guevara Noubir. “Single-stroke language-agnostic keylogging using stereo-microphones and domain specific machine learning”. In: *Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks - WiSec ’14*. Ed. by Andrew Martin et al. New York, New York, USA: ACM Press, 2014, pp. 201–212. DOI: 10.1145/2627393.2627417.
- [276] Sashank Narain et al. “Inferring User Routes and Locations Using Zero-Permission Mobile Sensors”. In: *2016 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2016, pp. 397–413. DOI: 10.1109/sp.2016.31.
- [277] Sashank Narain et al. “The Perils of User Tracking Using Zero-Permission Mobile Apps”. In: *IEEE Security and Privacy Magazine* 15.2 (2017), pp. 32–41. DOI: 10.1109/msp.2017.25.
- [278] Kalan Nawarathne et al. “Dead reckoning on smartphones to reduce GPS usage”. In: *2014 13th International Conference on Control Automation Robotics & Vision (ICARCV)*. IEEE, 2014, pp. 529–534. DOI: 10.1109/icarcv.2014.7064360.
- [279] Sarfraz Nawaz and Cecilia Mascolo. “Mining users’ significant driving routes with low-power sensors”. In: *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*. Ed. by Ákos Lédecz, Prabal Dutta, and Chenyang Lu. New York, NY, USA: Acm, 2014, pp. 236–250. DOI: 10.1145/2668332.2668348.
- [280] Rui Ning et al. “DeepMag: Sniffing Mobile Apps in Magnetic Field through Deep Convolutional Neural Networks”. In: *2018 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. IEEE, 2018, pp. 1–10. DOI: 10.1109/percom.2018.8444573.

- [281] Helen Nissenbaum. “A Contextual Approach to Privacy Online”. In: *Daedalus* 140.4 (2011), pp. 32–48. DOI: 10.1162/DAED_a_00113.
- [282] Mirja Nitschke et al. “Harmonized Group Mix for ITS”. In: *Proceedings of the 6th International Conference on Information Systems Security and Privacy*. SCITEPRESS - Science and Technology Publications, 2020, pp. 152–163. DOI: 10.5220/0008989101520163.
- [283] Mirja Nitschke et al. “Harmonic Group Mix: A Framework for Anonymous and Authenticated Broadcast Messages in Vehicle-to-Vehicle Environments”. In: *Information Systems Security and Privacy*. Ed. by Steven Furnell et al. Vol. 1545. Communications in Computer and Information Science. Cham: Springer International Publishing, 2022, pp. 178–200. DOI: 10.1007/978-3-030-94900-6_9.
- [284] Patrica A. Norberg, Daniel R. Horne, and David A. Horne. “The Privacy Paradox: Personal Information Disclosure Intentions versus Behaviors”. In: *Journal of Consumer Affairs* 41.1 (2007), pp. 100–126. DOI: 10.1111/j.1745-6606.2006.00070.x.
- [285] T. O’Kane and J. V. Ringwood. “Vehicle Speed Estimation Using GPS/RISS (Reduced Inertial Sensor System)”. In: *24th IET Irish Signals and Systems Conference (ISSC 2013)*. Institution of Engineering and Technology, 2013, p. 43. DOI: 10.1049/ic.2013.0046.
- [286] Alan V. Oppenheim, John R. Buck, and Ronald W. Schafer. *Discrete-time signal processing*. 2. ed., internat. ed. Prentice Hall signal processing series. Upper Saddle River, NJ: Prentice-Hall Internat, 1999.
- [287] George Orwell. *Nineteen Eighty-Four*. London: Penguin Books, 1990.
- [288] Emmanuel Owusu et al. “ACcessory”. In: *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications - HotMobile ’12*. Ed. by Gaetano Borriello and Rajesh Krishna Balan. New York, New York, USA: ACM Press, 2012, p. 1. DOI: 10.1145/2162081.2162095.
- [289] Pascal Paillier and David Pointcheval. “Efficient Public-Key Cryptosystems Provably Secure Against Active Adversaries”. In: *Advances in Cryptology - ASIACRYPT’99*. Ed. by Gerhard Goos et al. Vol. 1716. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 165–179. DOI: 10.1007/978-3-540-48000-6_14.
- [290] Candy Pang and Duane Szafron. “Single Source of Truth (SSOT) for Service Oriented Architecture (SOA)”. In: *Service-Oriented Computing*. Ed. by Xavier Franch et al. Vol. 8831. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 575–589. DOI: 10.1007/978-3-662-45391-9_50.
- [291] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

- [292] Ken Peffers et al. “A Design Science Research Methodology for Information Systems Research”. In: *Journal of Management Information Systems* 24.3 (2007), pp. 45–77. DOI: 10.2753/mis0742-1222240302.
- [293] Beatrice Perez, Mirco Musolesi, and Gianluca Stringhini. “Fatal attraction”. In: *Proceedings of the 12th Conference on Security and Privacy in Wireless and Mobile Networks*. New York, NY, USA: Acm, 2019, pp. 163–173. DOI: 10.1145/3317549.3319726.
- [294] Mert D. Pesé, Xiaoying Pu, and Kang G. Shin. “SPy: Car Steering Reveals Your Trip Route!” In: *Proceedings on Privacy Enhancing Technologies* 2020.2 (2020), pp. 155–174. DOI: 10.2478/popets-2020-0022.
- [295] Jonathan Petit et al. “Pseudonym Schemes in Vehicular Networks: A Survey”. In: *IEEE Communications Surveys & Tutorials* 17.1 (2015), pp. 228–255. DOI: 10.1109/comst.2014.2345420.
- [296] François Petitjean, Alain Ketterlin, and Pierre Gançarski. “A global averaging method for dynamic time warping, with applications to clustering”. In: *Pattern Recognition* 44.3 (2011), pp. 678–693. DOI: 10.1016/j.patcog.2010.09.013.
- [297] A. Pfitzmann et al. “Trusting mobile user devices and security modules”. In: *Computer* 30.2 (1997), pp. 61–68. DOI: 10.1109/2.566159.
- [298] Joseph Phelps, Glen Nowak, and Elizabeth Ferrell. “Privacy Concerns and Consumer Willingness to Provide Personal Information”. In: *Journal of Public Policy & Marketing* 19.1 (2000), pp. 27–41. DOI: 10.1509/jppm.19.1.27.16941.
- [299] Dan Ping, Xin Sun, and Bing Mao. “TextLogger”. In: *Proceedings of the 8th ACM Conference on Security & Privacy in Wireless and Mobile Networks*. New York, NY, USA: Acm, 2015, pp. 1–12. DOI: 10.1145/2766498.2766511.
- [300] Siim Plangi et al. “Real-Time Vehicles Tracking Based on Mobile Multi-Sensor Fusion”. In: *IEEE Sensors Journal* 18.24 (2018), pp. 10077–10084. DOI: 10.1109/jsen.2018.2873050.
- [301] Joachim Plesch and Irenaeus Wolff. “Personal-Data Disclosure in a Field Experiment: Evidence on Explicit Prices, Political Attitudes, and Privacy Preferences”. In: *Games* 9.2 (2018), p. 24. DOI: 10.3390/g9020024.
- [302] Android Open Source Project. *Sensors Types*. 2020. URL: <https://source.android.com/devices/sensors/sensor-types>.
- [303] Emmanuel Prouff and Matthieu Rivain. “Masking against Side-Channel Attacks: A Formal Security Proof”. In: (2013), pp. 142–159. URL: <https://doi.org/10.1007%5C%2F978-3-642-38348-9%5C%5F9>.
- [304] Zhan Fan Quek and Eldwin Ng. *Driver Identification by Driving Style*. 2013.
- [305] Julien Rabatel, Sandra Bringay, and Pascal Poncelet. “Anomaly detection in monitoring sensor data for preventive maintenance”. In: *Expert Systems with Applications* 38.6 (2011), pp. 7003–7015. DOI: 10.1016/j.eswa.2010.12.014.

- [306] Kasturi Rangan Raghavan et al. “Override”. In: *Proceedings of the Third International Workshop on Sensing Applications on Mobile Phones - PhoneSense '12*. Ed. by David Chu and Ramesh Govindan. New York, New York, USA: ACM Press, 2012, pp. 1–5. DOI: 10.1145/2389148.2389150.
- [307] Kai Rannenberg, Jan Camenisch, and Ahmad Sabouri. *Attribute-based Credentials for Trust*. Cham: Springer International Publishing, 2015. DOI: 10.1007/978-3-319-14439-9.
- [308] Siegfried Rasthofer et al. “DroidForce: Enforcing Complex, Data-centric, System-wide Policies in Android”. In: *2014 Ninth International Conference on Availability, Reliability and Security*. IEEE, 2014, pp. 40–49. DOI: 10.1109/ares.2014.13.
- [309] *Richtlinien für die Sicherung von Arbeitsstellen an Strassen: (RSA)*. 6. überarbeitete Auflage. Bonn: Kirschbaum, 2017.
- [310] *Richtlinien für Lichtsignalanlagen: RiLSA : Lichtzeichenanlagen für den StraSSenverkehr*. Köln, 2015.
- [311] Cory Robinson. “Disclosure of personal data in ecommerce: A cross-national comparison of Estonia and the United States”. In: *Telematics and Informatics* 34.2 (2017), pp. 569–582. DOI: 10.1016/j.tele.2016.09.006.
- [312] Bjoern Roeber et al. “Personal data: how context shapes consumers’ data sharing with organizations from various sectors”. In: *Electronic Markets* 25.2 (2015), pp. 95–108. DOI: 10.1007/s12525-015-0183-0.
- [313] Roman Schlegel et al. “Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones”. In: *Proceedings of the Network and Distributed System Security Symposium*. San Diego, California, USA: The Internet Society, Feb. 2011.
- [314] Cristina Romero-Tris and David Megías. “Protecting Privacy in Trajectories with a User-Centric Approach”. In: *ACM Transactions on Knowledge Discovery from Data* 12.6 (2018), pp. 1–27. DOI: 10.1145/3233185.
- [315] Hes Ronald and John J Borking, eds. *Privacy-enhancing technologies: The path to anonymity*. Rev. ed. Vol. 11. Achtergrondstudies en verkenningen / Registratiekamer. TheHague: Registratiekamer, 1998.
- [316] Christian Roth, Lukas Hartmann, and Doan Kesdoan. “PARTS – Privacy-Aware Routing with Transportation Subgraphs”. In: *Secure IT Systems*. Ed. by Helger Lipmaa, Aikaterini Mitrokotsa, and Raimundas Matulevicius. Vol. 10674. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2017, pp. 86–101. DOI: 10.1007/978-3-319-70290-2_6.
- [317] Christian Roth and Dogan Kesdogan. “A Privacy Enhanced Crowdsourcing Architecture for Road Information Mining Using Smartphones”. In: *2018 IEEE 11th Conference on Service-Oriented Computing and Applications (SOCA)*. IEEE, 2018, pp. 17–24. DOI: 10.1109/soca.2018.00010.

- [318] Christian Roth, Mario Saur, and Dogan Kesdogan. “kUBI: A Framework for Privacy and Transparency in Sensor-Based Business Models for Consumers: A Pay-How-You-Drive Example”. In: *Computer Security*. Ed. by Ioana Boureau et al. Vol. 12580. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 114–132. DOI: 10.1007/978-3-030-66504-3_7.
- [319] Christian Roth, Ngoc Thanh Dinh, and Dogan Kesdogan. “CrowdAbout: Using Vehicles as Sensors to Improve Map Data for ITS”. In: *2020 Seventh International Conference on Social Networks Analysis, Management and Security (SNAMS)*. IEEE, 2020, pp. 1–8. DOI: 10.1109/snams52053.2020.9336531.
- [320] Christian Roth et al. “Dynamische Teilrouten zur anonymen Navigation”. In: *Fachgespräch Ortsbezogene Anwendungen und Dienste*. 2016. URL: <https://epub.uni-regensburg.de/36336/> (visited on 12/22/2021).
- [321] Christian Roth et al. “My Smartwatch Is Mine – Machine Learning Based Theft Detection of Smartwatches”. In: *Secure IT Systems*. Ed. by Aslan Askarov, René Rydhof Hansen, and Willard Rafnsson. Vol. 11875. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 171–187. DOI: 10.1007/978-3-030-35055-0_11.
- [322] Christian Roth et al. “Are Sensor-Based Business Models a Threat to Privacy? The Case of Pay-How-You-Drive Insurance Models”. In: *Trust, Privacy and Security in Digital Business*. Ed. by Stefanos Gritzalis et al. Vol. 12395. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 75–85. DOI: 10.1007/978-3-030-58986-8_6.
- [323] Christian Roth et al. “iTLM: A Privacy Friendly Crowdsourcing Architecture for Intelligent Traffic Light Management”. In: *Proceedings of the 9th International Conference on Data Science, Technology and Applications*. SCITEPRESS - Science and Technology Publications, 2020, pp. 252–259. DOI: 10.5220/0009831902520259.
- [324] Christian Roth et al. “DaRoute: Inferring trajectories from zero-permission smartphone sensors”. In: *2021 18th International Conference on Privacy, Security and Trust (PST)*. IEEE, 2021, pp. 1–10. DOI: 10.1109/pst52912.2021.9647811.
- [325] Christian Roth et al. “iTLM-Q: A Constraint-Based Q-Learning Approach for Intelligent Traffic Light Management”. In: *Data Management Technologies and Applications*. Ed. by Slimane Hammoudi, Christoph Quix, and Jorge Bernardino. Vol. 1446. Communications in Computer and Information Science. Cham: Springer International Publishing, 2021, pp. 56–79. DOI: 10.1007/978-3-030-83014-4_3.
- [326] Christian Roth et al. “ROADR: towards road network assessment using everyone-as-a-sensor”. In: *The 2nd International Conference on Distributed Sensing and Intelligent Systems (ICDISIS 2021)*. Institution of Engineering and Technology, 2021, pp. 28–39. DOI: 10.1049/icp.2021.2660.

- [327] Christian Roth et al. “STRIDE: Secure Traffic Reporting Infrastructure based on Distributed Entities”. In: *2021 Sixth International Conference on Fog and Mobile Edge Computing (FMEC)*. IEEE, 2021, pp. 1–6. DOI: 10.1109/FMEC54266.2021.9732577.
- [328] Peter J. Rousseeuw. “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis”. In: *Journal of Computational and Applied Mathematics* 20 (1987), pp. 53–65. DOI: 10.1016/0377-0427(87)90125-7.
- [329] Michele Ruta et al. “Semantic annotation of OpenStreetMap points of interest for mobile discovery and navigation”. In: *Proceedings - 2012 IEEE 1st International Conference on Mobile Services, MS 2012*. 2012, pp. 33–39.
- [330] Nazir Saleheen et al. “mSieve: Differential Behavioral Privacy in Time Series of Mobile Sensor Data”. In: *Proceedings of the ... ACM International Conference on Ubiquitous Computing . UbiComp (Conference) 2016 (2016)*, pp. 706–717. DOI: 10.1145/2971648.2971753.
- [331] E. Sarasketa-Zabala et al. “Realistic lifetime prediction approach for Li-ion batteries”. In: *Applied Energy* 162 (2016), pp. 839–852. DOI: 10.1016/j.apenergy.2015.10.115.
- [332] Florian Marcus Schaub. “Dynamic privacy adaptation in ubiquitous computing”. PhD thesis. Universität Ulm, 2014. DOI: 10.18725/oparu-3188.
- [333] Albrecht Schmidt et al. “Advanced Interaction in Context”. In: *Handheld and Ubiquitous Computing*. Ed. by Gerhard Goos et al. Vol. 1707. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 89–101. DOI: 10.1007/3-540-48157-5_10.
- [334] Emanuel Schmitt and Jan-Niklas Voigt-Antons. “Predicting Tap Locations on Touch Screens in the Field Using Accelerometer and Gyroscope Sensor Readings”. In: *HCI for Cybersecurity, Privacy and Trust*. Ed. by Abbas Moallem. Vol. 12210. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 637–651. DOI: 10.1007/978-3-030-50309-3_43.
- [335] Simeon Schudy and Verena Utikal. “‘You must not know about me’—On the willingness to share personal data”. In: *Journal of Economic Behavior & Organization* 141 (2017), pp. 1–13. DOI: 10.1016/j.jebo.2017.05.023.
- [336] Salahadin Seid et al. “Mobile Crowdsensing Based Road Surface Monitoring Using Smartphone Vibration Sensor and Lorawan”. In: *Proceedings of the 1st Workshop on Experiences with the Design and Implementation of Frugal Smart Objects*. New York, NY, USA: Acm, 2020, pp. 36–41. DOI: 10.1145/3410670.3410858.
- [337] Harit Sharma et al. “S-road assist: Road surface conditions and driving behavior analysis using smartphones”. In: *2015 International Conference on Connected Vehicles and Expo (ICCVE)*. IEEE, 2015, pp. 291–296. DOI: 10.1109/iccve.2015.61.

- [338] Chao Shen et al. “Input extraction via motion-sensor behavior analysis on smartphones”. In: *Computers & Security* 53 (2015), pp. 143–155. DOI: 10.1016/j.cose.2015.06.013.
- [339] Chao Shen et al. “On motion sensors as source for user input inference in smartphones”. In: *IEEE International Conference on Identity, Security and Behavior Analysis (ISBA 2015)*. IEEE, 2015, pp. 1–6. DOI: 10.1109/isba.2015.7126368.
- [340] Yiran Shen et al. “Privacy-preserving sparse representation classification in cloud-enabled mobile applications”. In: *Computer Networks* 133 (2018), pp. 59–72. DOI: 10.1016/j.comnet.2018.01.035.
- [341] Amit Kumar Sikder, Hidayet Aksu, and A. Selcuk Uluagac. “6thSense: A Context-aware Sensor-based Attack Detector for Smart Devices”. In: *26th USENIX Security Symposium (USENIX Security 17)*. Vancouver, BC: USENIX Association, 2017, pp. 397–414. URL: <https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/sikder>.
- [342] Pekka Sillberg et al. “Challenges in the Interpretation of Crowdsourced Road Condition Data”. In: *2018 International Conference on Intelligent Systems (IS)*. IEEE, 2018, pp. 215–221. DOI: 10.1109/is.2018.8710571.
- [343] Nuno Silva et al. “Anomaly Detection in Roads with a Data Mining Approach”. In: *Procedia Computer Science* 121 (2017), pp. 415–422. DOI: 10.1016/j.procs.2017.11.056.
- [344] Herbert A. Simon. “Bounded Rationality”. In: *The new Palgrave: utility and probability*. Ed. by John Eatwell and Murray Milgate. London: Macmillan, 1990, pp. 15–18. DOI: 10.1007/978-1-349-20568-4_5.
- [345] Laurent Simon and Ross Anderson. “PIN skimmer”. In: *Proceedings of the Third ACM workshop on Security and privacy in smartphones & mobile devices - SPSM '13*. Ed. by William Enck, Adrienne Porter Felt, and N. Asokan. New York, New York, USA: ACM Press, 2013, pp. 67–78. DOI: 10.1145/2516760.2516770.
- [346] Gurdit Singh et al. “Smart patrolling: An efficient road surface monitoring using smartphone sensors and crowdsourcing”. In: *Pervasive and Mobile Computing* 40 (2017), pp. 71–88. DOI: 10.1016/j.pmcj.2017.06.002.
- [347] Vikrant Singh et al. “SafeStreet: An automated road anomaly detection and early-warning system using mobile crowdsensing”. In: *2018 10th International Conference on Communication Systems and Networks, COMSNETS 2018*. Vol. 2018-January. Institute of Electrical and Electronics Engineers Inc, 2018, pp. 549–552.
- [348] Smith, Dinev, and Xu. “Information Privacy Research: An Interdisciplinary Review”. In: *MIS Quarterly* 35.4 (2011), p. 989. DOI: 10.2307/41409970.
- [349] Daniel J. Solove. “A Taxonomy of Privacy”. In: *University of Pennsylvania Law Review* 154.3 (2006), p. 477. DOI: 10.2307/40041279.

- [350] Daniel J. Solove. “‘I’ve Got Nothing to Hide’ and Other Misunderstandings of Privacy”. In: *San Diego Law Reviews* 44 (2007), pp. 745–772. URL: <https://scholarlycommons.law.wlu.edu/wlulr/vol25/iss1/20/>.
- [351] Daniel J. Solove. “The Myth of the Privacy Paradox”. In: *SSRN Electronic Journal* (2020). DOI: 10.2139/ssrn.3536265.
- [352] Rui Song et al. “WebLogger: Stealing your personal PINs via mobile web application”. In: *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*. IEEE, 2017, pp. 1–6. DOI: 10.1109/wcsp.2017.8171036.
- [353] Rui Song et al. “I Know What You Type: Leaking User Privacy via Novel Frequency-Based Side-Channel Attacks”. In: *2018 IEEE Global Communications Conference (GLOBECOM)*. Piscataway, NJ: IEEE, 2018, pp. 1–6. DOI: 10.1109/glocom.2018.8647385.
- [354] Vinicius M.A. Souza, Rafael Giusti, and Antônio J.L. Batista. “Asfalt: A low-cost system to evaluate pavement conditions in real-time using smartphones and machine learning”. In: *Pervasive and Mobile Computing* 51 (2018), pp. 121–137. DOI: 10.1016/j.pmcj.2018.10.008.
- [355] Raphael Spreitzer. “PIN Skimming”. In: *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices*. Ed. by Cliff Wang et al. New York, NY, USA: Acm, 2014, pp. 51–62. DOI: 10.1145/2666620.2666622.
- [356] Raphael Spreitzer et al. “Systematic Classification of Side-Channel Attacks: A Case Study for Mobile Devices”. In: *IEEE Communications Surveys & Tutorials* 20.1 (2018), pp. 465–488. DOI: 10.1109/comst.2017.2779824.
- [357] Marcin Staniek. “Road pavement condition diagnostics using smartphone-based data crowdsourcing in smart cities”. In: *Journal of Traffic and Transportation Engineering (English Edition)* 8.4 (2021), pp. 554–567. DOI: 10.1016/j.jtte.2020.09.004.
- [358] Bohua Sun et al. “Research on the Classification and Identification of Driver’s Driving Style”. In: *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*. IEEE, 2017, pp. 28–32. DOI: 10.1109/iscid.2017.31.
- [359] Yongqiang Sun et al. “Location information disclosure in location-based social network services: Privacy calculus, benefit structure, and gender differences”. In: *Computers in Human Behavior* 52 (2015), pp. 278–292. DOI: 10.1016/j.chb.2015.06.006.
- [360] Supriyo Chakraborty et al. “ipShield: A Framework For Enforcing Context-Aware Privacy”. In: *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*. Seattle, WA: USENIX Association, 2014, pp. 143–156. DOI: 10.5555/2616448.2616463.

- [361] James Surowiecki. *The wisdom of crowds: Why the many are smarter than the few and how collective wisdom shapes business economies societies and nations*. New York: Doubleday, 2004.
- [362] Daniel Susser, Beate Roessler, and Helen Nissenbaum. “Technology, autonomy, and manipulation”. In: *Internet Policy Review* 8.2 (2019). DOI: 10.14763/2019.2.1410.
- [363] Tracey Swift. “Trust, reputation and corporate accountability to stakeholders”. In: *Business Ethics: A European Review* 10 (2001), pp. 16–26. DOI: 10.1111/1467-8608.00208.
- [364] B. Syed et al. “A smart transport application of cyber-physical systems: Road surface monitoring with mobile devices”. In: *2012 Sixth International Conference on Sensing Technology (ICST)*. IEEE, 2012, pp. 8–12. DOI: 10.1109/ICSensT.2012.6461796.
- [365] Fatemeh Tahmasbi et al. “Your Phone Tells Us The Truth”. In: *Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. Ed. by Rajeev Shorey et al. New York, NY, USA: Acm, Oct. 2018, pp. 762–764. DOI: 10.1145/3241539.3267769. URL: <https://doi.org/10.1145/3241539.3267769>.
- [366] Benxiao Tang et al. “Niffler: A Context-Aware and User-Independent Side-Channel Attack System for Password Inference”. In: *Wireless Communications and Mobile Computing* 2018 (2018), pp. 1–19. DOI: 10.1155/2018/4627108.
- [367] Hui Yie Teh, Andreas W. Kempa-Liehr, and Kevin I-Kai Wang. “Sensor data quality: a systematic review”. In: *Journal of Big Data* 7.1 (Feb. 2020). DOI: 10.1186/s40537-020-0285-1. URL: <https://doi.org/10.1186/s40537-020-0285-1>.
- [368] The Insight Partners and Statista. *Projected global automotive sensor market size 2025*. 2020. URL: <https://www.statista.com/statistics/1011203/projected-global-automotive-sensor-market/>.
- [369] Saurabh Tiwari, Ravi Bhandari, and Bhaskaran Raman. “RoadCare”. In: *Proceedings of the 3rd ACM SIGCAS Conference on Computing and Sustainable Societies*. New York, NY, USA: Acm, 2020, pp. 231–242. DOI: 10.1145/3378393.3402284.
- [370] Jolanta Tober et al. *Ein ADAC-Leitfaden fuer die Praxis*. 2005.
- [371] Tobias Fiebig, Jan Krissler, and Ronny Hänsch. “Security Impact of High Resolution Smartphone Cameras”. In: 2014. URL: <https://www.usenix.org/conference/woot14/workshop-program/presentation/fiebig>.
- [372] Sabine Trepte et al. “A Cross-Cultural Perspective on the Privacy Calculus”. In: *Social Media + Society* 3.1 (2017), p. 205630511668803. DOI: 10.1177/2056305116688035.

- [373] Carmela Troncoso et al. “PriPAYD: Privacy-Friendly Pay-As-You-Drive Insurance”. In: *IEEE Transactions on Dependable and Secure Computing* 8.5 (2011), pp. 742–755. DOI: 10.1109/tdsc.2010.71.
- [374] Dimitrios I. Tselentis, George Yannis, and Eleni I. Vlahogianni. “Innovative Insurance Schemes: Pay as/how You Drive”. In: *Transportation Research Procedia* 14 (2016), pp. 362–371. DOI: 10.1016/j.trpro.2016.05.088.
- [375] Vijay K. Vaishnavi. *Design Science Research Methods and Patterns*. Auerbach Publications, 2007. DOI: 10.1201/9781420059335.
- [376] Minh van Ly, Sujitha Martin, and Mohan M. Trivedi. “Driver classification and driving style recognition using inertial sensors”. In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2013, pp. 1040–1045. DOI: 10.1109/ivs.2013.6629603.
- [377] Jeroen van Rest et al. “Designing Privacy-by-Design”. In: *Privacy Technologies and Policy*. Ed. by David Hutchison et al. Vol. 8319. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 55–72. DOI: 10.1007/978-3-642-54069-1_4.
- [378] Projektgruppe der Arbeitsgemeinschaft Wirtschaftlicher Verbraucherschutz. *Telematiktarife im Versicherungsbereich Abschlussbericht der Projektgruppe der Arbeitsgemeinschaft Wirtschaftlicher Verbraucherschutz*. Tech. rep. 2019.
- [379] Eric Verheul, Christopher Hicks, and Flavio D. Garcia. “IFAL: Issue First Activate Later Certificates for V2X”. In: *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 2019, pp. 279–293. DOI: 10.1109/EuroSP.2019.00029.
- [380] Rohit Verma et al. “Smart-phone based Spatio-temporal Sensing for Annotated Transit Map Generation”. In: *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. Ed. by Erik Hoel et al. New York, NY, USA: Acm, 2017, pp. 1–10. DOI: 10.1145/3139958.3140005.
- [381] Nuttun Virojboonkiate, Peerapon Vateekul, and Kultida Rojviboonchai. “Driver identification using histogram and neural network from acceleration data”. In: *2017 IEEE 17th International Conference on Communication Technology (ICCT)*. IEEE, 2017, pp. 1560–1564. DOI: 10.1109/icct.2017.8359893.
- [382] Astarita Vittorio et al. “Automated Sensing System for Monitoring of Road Surface Quality by Mobile Devices”. In: *Procedia - Social and Behavioral Sciences* 111 (2014), pp. 242–251. DOI: 10.1016/j.sbspro.2014.01.057.
- [383] W3c. *DeviceOrientation Event Specification*. 2011. URL: <https://www.w3.org/TR/2011/WD-orientation-event-20111201/>.
- [384] W3c. *Accelerometer*. 2021. URL: <https://www.w3.org/TR/accelerometer/>.
- [385] W3c. *Gyroscope*. 2021. URL: <https://www.w3.org/TR/gyroscope/>.

- [386] W3c. *Magnetometer*. 2021. URL: <https://www.w3.org/TR/magnetometer/>.
- [387] Lisa-Marie Wadle, Noemi Martin, and Daniel Ziegler. “Privacy and Personalization”. In: *Adjunct Publication of the 27th Conference on User Modeling, Adaptation and Personalization*. Ed. by George Angelos Papadopoulos et al. New York, NY, USA: Acm, 2019, pp. 319–324. DOI: 10.1145/3314183.3323672.
- [388] Ari Ezra Waldman. “Cognitive biases, dark patterns, and the ‘privacy paradox’”. In: *Current Opinion in Psychology* 31 (2020), pp. 105–109. DOI: 10.1016/j.copsyc.2019.08.025.
- [389] Marian Waltereit, Maximilian Uphoff, and Torben Weis. “Route Derivation Using Distances and Turn Directions”. In: *Proceedings of the ACM Workshop on Automotive Cybersecurity*. Ed. by Ziming Zhao, Qi Alfred Chen, and Gail-Joon Ahn. New York, NY, USA: Acm, 2019, pp. 35–40. DOI: 10.1145/3309171.3309176.
- [390] Huaijun Wang et al. “A Road Quality Detection Method Based on the Mahalanobis-Taguchi System”. In: *IEEE Access* 6 (2018), pp. 29078–29087. DOI: 10.1109/access.2018.2839765.
- [391] Qianlong Wang et al. “DeepSpeedometer: Vehicle speed estimation from accelerometer and gyroscope using LSTM model”. In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–6. DOI: 10.1109/itsc.2017.8317935.
- [392] WenChen Wang et al. “Road Anomaly Detection with Group Intelligence Perception”. In: *2019 International Conference on Communications, Information System and Computer Engineering (CISCE)*. IEEE, 2019, pp. 678–682. DOI: 10.1109/cisce.2019.00157.
- [393] Yan Wang et al. “Sensing vehicle dynamics for determining driver phone use”. In: *Proceeding of the 11th annual international conference on Mobile systems, applications, and services - MobiSys '13*. Ed. by Hao-Hua Chu et al. New York, New York, USA: ACM Press, 2013, p. 41. DOI: 10.1145/2462456.2464447.
- [394] Yao Wang et al. “GazeRevealer”. In: *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. Ed. by Henning Schulzrinne and Pan Li. New York, NY, USA: Acm, 2018, pp. 254–263. DOI: 10.1145/3286978.3287026.
- [395] Samuel D. Warren and Louis D. Brandeis. “The Right to Privacy”. In: *Harvard Law Review* 4.5 (1890), pp. 193–220.
- [396] Maor Weinberger, Dan Bouhnik, and Maayan Zhitomirsky-Geffet. “Factors Affecting Students’ Privacy Paradox and Privacy Protection Behavior”. In: *Open Information Science* 1.1 (2017). DOI: 10.1515/opis-2017-0002.
- [397] Mark Weiser. “The Computer for the 21st Century”. In: *Scientific American* 265.3 (1991), pp. 94–104. DOI: 10.1038/scientificamerican0991-94.

- [398] Alan F. Westin. “Privacy And Freedom”. In: (1967). URL: <https://scholarlycommons.law.wlu.edu/cgi/viewcontent.cgi?article=3659%5C&context=wlulr>.
- [399] Alan F. Westin. “Social and Political Dimensions of Privacy”. In: *Journal of Social Issues* 59.2 (2003), pp. 431–453. DOI: 10.1111/1540-4560.00072.
- [400] Valentine Weydert, Pierre Desmet, and Caroline Lancelot-Miltgen. “Convincing consumers to share personal data: double-edged effect of offering money”. In: *Journal of Consumer Marketing* 37.1 (2019), pp. 1–9. DOI: 10.1108/jcm-06-2018-2724.
- [401] William Whyte et al. “A security credential management system for V2V communications”. In: *2013 IEEE Vehicular Networking Conference*. IEEE, 2013, pp. 1–8. DOI: 10.1109/vnc.2013.6737583.
- [402] Thomas Wilde and Thomas Hess. “Forschungsmethoden der Wirtschaftsinformatik”. In: *Wirtschaftsinformatik* 49.4 (2007), pp. 280–287. DOI: 10.1007/s11576-007-0064-z.
- [403] Myounggyu Won, Ashutosh Mishra, and Sang H. Son. “HybridBaro: Mining Driving Routes Using Barometer Sensor of Smartphone”. In: *IEEE Sensors Journal* 17.19 (2017), pp. 6397–6408. DOI: 10.1109/jsen.2017.2734919.
- [404] Myounggyu Won et al. “Enabling energy-efficient driving route detection using built-in smartphone barometer sensor”. In: *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2016, pp. 2378–2385. DOI: 10.1109/itsc.2016.7795939.
- [405] Haiqin Wu et al. “Enabling Data Trustworthiness and User Privacy in Mobile Crowdsensing”. In: *IEEE/ACM Transactions on Networking* 27.6 (2019), pp. 2294–2307. DOI: 10.1109/tnet.2019.2944984.
- [406] Li-jun Wu. “Experimental study on vehicle speed estimation using accelerometer and wheel speed measurements”. In: *2011 Second International Conference on Mechanic Automation and Control Engineering*. IEEE, 2011, pp. 294–297. DOI: 10.1109/mace.2011.5986916.
- [407] Christian Wurmer, Christian Roth, and Dogan Kesdogan. “Providing AID: A Unifying Survey on the Privacy Paradox and Information Disclosure”. In: *ACM Computing Surveys (submitted)*.
- [408] Heng Xu et al. “Information Privacy Concerns: Linking Individual Perceptions with Institutional Privacy Assurances”. In: *Journal of the Association for Information Systems* 12.12 (2011), pp. 798–824. DOI: 10.17705/1jais.00281.
- [409] Zhi Xu, Kun Bai, and Sencun Zhu. “TapLogger”. In: *Proceedings of the fifth ACM conference on Security and Privacy in Wireless and Mobile Networks - WISEC '12*. Ed. by Marwan Krunz et al. New York, New York, USA: ACM Press, 2012, p. 113. DOI: 10.1145/2185448.2185465.

- [410] Zhi Xu and Sencun Zhu. “SemaDroid”. In: *Proceedings of the 5th ACM Conference on Data and Application Security and Privacy*. Ed. by Jaehong Park and Anna Squicciarini. New York, NY, USA: Acm, 2015, pp. 61–72. DOI: 10.1145/2699026.2699114.
- [411] Kai Xue, Tomonori Nagayama, and Boyu Zhao. “Road profile estimation and half-car model identification through the automated processing of smartphone data”. In: *Mechanical Systems and Signal Processing* 142 (2020), p. 106722. DOI: 10.1016/j.ymsp.2020.106722.
- [412] Yan Michalevsky, Dan Boneh, and Gabi Nakibly. “Gyrophone: Recognizing Speech from Gyroscope Signals”. In: *Proceedings of the Seventeenth Large Installation Systems Administration Conference (LISA XVII)*. Berkeley, Calif.: USENIX Association, 2003, pp. 1053–1067. URL: <https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/michalevsky>.
- [413] Ching-Han Yang, Deron Liang, and Chin-Chun Chang. “A novel driver identification method using wearables”. In: *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2016, pp. 1–5. DOI: 10.1109/ccnc.2016.7444722.
- [414] Qin Yanjun et al. “Transportation Mode Recognition Algorithm Based on Bayesian Voting”. In: *Industrial digitalization by enterprise systems*. Ed. by Zhibo Pang, Lefei Li, and Gang Li. Piscataway, NJ: IEEE, 2017, pp. 260–269. DOI: 10.1109/es.2017.50.
- [415] Ching-Hsuan Yeh et al. “What drives internet users’ willingness to provide personal information?” In: *Online Information Review* 42.6 (2018), pp. 923–939. DOI: 10.1108/oir-09-2016-0264.
- [416] Chih-Wei Yi, Yi-Ta Chuang, and Chia-Sheng Nian. “Toward Crowdsourcing-Based Road Pavement Monitoring by Mobile Sensing Technologies”. In: *IEEE Transactions on Intelligent Transportation Systems* 16.4 (2015), pp. 1905–1917. DOI: 10.1109/tits.2014.2378511.
- [417] Jiadi Yu et al. “SenSpeed: Sensing Driving Conditions to Estimate Vehicle Speed in Urban Environments”. In: *IEEE Transactions on Mobile Computing* 15.1 (2016), pp. 202–216. DOI: 10.1109/tmc.2015.2411270.
- [418] Yuebin Yu and Haorong Li. “Virtual in-situ calibration method in building systems”. In: *Automation in Construction* 59 (2015), pp. 59–67. DOI: 10.1016/j.autcon.2015.08.003.
- [419] Irving M. Zeitlin and George Caspar Homans. “Social Behavior: Its Elementary Forms”. In: *Social Forces* 54.2 (1975), p. 474. DOI: 10.2307/2576649.
- [420] Mattia Zeni, Enrico Bignotti, and Fausto Giunchiglia. “Combining Crowdsourcing and Crowdsensing to Infer the Spatial Context”. In: *2018 IEEE International Conference on Pervasive Computing and Communications workshops (PerCom workshops)*. Piscataway, NJ: IEEE, 2018, pp. 28–33. DOI: 10.1109/percomw.2018.8480312.

- [421] Cheng Zhang et al. “Driver Classification Based on Driving Behaviors”. In: *Proceedings of the 21st International Conference on Intelligent User Interfaces*. Ed. by Jeffrey Nichols et al. New York, NY, USA: Acm, 2016, pp. 80–84. DOI: 10.1145/2856767.2856806.
- [422] Chuan Zhang et al. “An efficient and privacy-preserving truth discovery scheme in crowdsensing applications”. In: *Computers & Security* 97 (2020), p. 101848. DOI: 10.1016/j.cose.2020.101848.
- [423] Jiexin Zhang, Alastair R. Beresford, and Ian Sheret. “SensorID: Sensor Calibration Fingerprinting for Smartphones”. In: *2019 IEEE Symposium on Security and Privacy*. Piscataway, NJ: IEEE, 2019, pp. 638–655. DOI: 10.1109/sp.2019.00072.
- [424] Kao Zhao et al. “Privacy Protection for Perceptual Applications on Smartphones”. In: *2015 IEEE International Conference on Mobile Services*. IEEE, 2015, pp. 174–181. DOI: 10.1109/MobServ.2015.33.
- [425] Xiaofeng Zhao et al. “CPDM: An Efficient Crowdsensing-Based Pothole Detection and Measurement System Design”. In: *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 2019, pp. 547–554. DOI: 10.1109/ictai.2019.00082.
- [426] Huadi Zheng and Haibo Hu. “MISSILE: A System of Mobile Inertial Sensor-Based Sensitive Indoor Location Eavesdropping”. In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 3137–3151. DOI: 10.1109/tifs.2019.2944034.
- [427] Lu Zhou et al. “Speed-Based Location Tracking in Usage-Based Automotive Insurance”. In: *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 2017, pp. 2252–2257. DOI: 10.1109/icdcs.2017.278.
- [428] Man Zhou et al. “PatternListener”. In: *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*. Ed. by David Lie et al. New York, NY, USA: Acm, 2018, pp. 1775–1787. DOI: 10.1145/3243734.3243777.
- [429] Zhe Zhou et al. “Acoustic Fingerprinting Revisited”. In: *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. Ed. by Gail-Joon Ahn, Moti Yung, and Ninghui Li. New York, NY, USA: Acm, 2014, pp. 429–440. DOI: 10.1145/2660267.2660300.
- [430] Martina Ziefle, Julian Halbey, and Sylvia Kowalewski. “Users’ Willingness to Share Data on the Internet: Perceived Benefits and Caveats”. In: *Proceedings of the International Conference on Internet of Things and Big Data*. SCITEPRESS - Science, 2016, pp. 255–265. DOI: 10.5220/0005897402550265.

Supervised

- [S1] Sebastian Aringer. “Identifying drivers using mobile sensor data”. Masters’s Thesis. University of Regensburg, Apr. 2019.
- [S2] Ngoc-Thanh Dinh. “Exploiting Zero-Permission Mobile Sensors for Vehicle Trajectory Inference”. Masters’s Thesis. University of Regensburg, May 2021.
- [S3] Michael Franz. “Entwicklung eines Algorithmus für die intelligente Erkennung von Baustellen anhand von Floating Phone Data”. Bachelor’s Thesis. University of Regensburg, Jan. 2020.
- [S4] Christina Gihl. “Ansätze zur Fahreridentifikation mithilfe von Sensordaten”. Seminar. University of Regensburg, Feb. 2019.
- [S5] Alice Grohmann. “Extraktion personenbezogener Informationen auf Basis von Smartphone-basierten Sensordaten: ein Literaturüberblick”. Bachelor’s Thesis. University of Regensburg, May 2020.
- [S6] Leon David Hagemann. “Privacy Enhancing Technologies. Eine technologische Übersicht und Einordnung”. Seminar. University of Regensburg, Feb. 2021.
- [S7] Sid Heiland. “Methoden zur sensorgetriebenen Streckenrückverfolgung”. Seminar. University of Regensburg, July 2020.
- [S8] Marco Faltermeier Jonas Trautner. “Abschätzung der Geschwindigkeit auf Basis von Smartphones - Eine Übersicht von Verfahren”. Seminar. University of Regensburg, July 2021.
- [S9] Marco Faltermeier Jonas Trautner. “Smartphone to Vehicle Alignment”. Seminar. University of Regensburg, Aug. 2021.
- [S10] Julia Karel. “Kategorisierung von sensorbasierten Seitenkanalangriffen auf Smartphones”. Seminar. University of Regensburg, Feb. 2021.
- [S11] Adrian Lanzl. “Real-Time Driving Speed Estimation Using Smartphone Motion Sensors”. Bachelor’s Thesis. University of Regensburg, Aug. 2020.
- [S12] Georg Schöll. “NoGPSNav Eine Lösung zur Navigation ohne die Verwendung externer Ortungs-Services”. Masters’s Thesis. University of Regensburg, Oct. 2020.
- [S13] Verena Schröppel. “Recognition of Traffic Lights using Floating Phone Data: Improving Machine Learning Performance with Feature Engineering”. Bachelor’s Thesis. University of Regensburg, Nov. 2020.
- [S14] Richard Schuster. “PETs and TETs for Sensor-Based Applications: A Scoping Review”. Seminar. University of Regensburg, Feb. 2021.

- [S15] Alexander Sheppard. "Overview of methods for determining road properties with smartphone sensors". Seminar. University of Regensburg, July 2021.
- [S16] Anita Wasserle. "Usage and Security Implications of Sensor Data Provided by Mobile Devices: A Systematic Literature Review". Masters's Thesis. University of Regensburg, May 2020.

Images

- [I1] Android Developer. *Install-time permissions*. Online; accessed September 24, 2021. 2021. URL: <https://developer.android.com/images/training/permissions/install-time.svg>.
- [I2] Android Developer. *One-time permissions*. Online; accessed September 24, 2021. 2021. URL: <https://developer.android.com/images/training/permissions/one-time-prompt.svg>.
- [I3] Android Developer. *Request app permissions*. Online; accessed September 24, 2021. 2021. URL: <https://developer.android.com/images/training/permissions/workflow-runtime.svg>.
- [I4] Android Developer. *Runtime permissions*. Online; accessed September 24, 2021. 2021. URL: <https://developer.android.com/images/training/permissions/runtime.svg>.
- [I5] HUK-COBURG Versicherungsgruppe. *Was ist Telematik*. Online; accessed September 06, 2021. 2021. URL: <https://www.huk.de/content/dam/hukde/images/uebersichtsseiten/standardteaser/telematik-1800.scale.sTeaserS2x.jpg>.