Institut für Parallele und Verteilte Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit

Schutz der Privatsphäre in verteilten Software-Defined-Car-Anwendungen

Dominik Held

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. rer. nat. habil. Holger Schwarz
Betreuer/in:	Dr. rer. nat. Pascal Hirmer
Beginn am:	15. Oktober 2021
Beendet am:	15. April 2021

Kurzfassung

Diese Bachelorarbeit setzt sich mit dem Thema der situationsabhängigen Anonymisierung von Daten, die von *Connected Cars* erzeugt und verarbeitet werden, auseinander.

Dabei wird mit PLEvS ein Konzept vorgestellt, mit dem sich Situationen modellieren und auswerten lassen. Anhand des Status der Situationen können Regeln definiert werden, durch welche sich der Anonymisierungsgrad der verarbeiteten Daten bestimmen lässt.

Des Weiteren wurden Algorithmen entworfen, mit denen sich konkrete Datentypen eines Connected Cars anonymisieren lassen, um die Privatsphäre von Benutzern zu schützen. Mit Lo*k*A lassen sich Ortsdaten anonymisieren. Der Algorithmus arbeitet mit Wahrscheinlichkeiten, um die größtmögliche Anonymität herzustellen. CaDaA befasst sich mit der Verschleierung von sensiblen Daten, die in den erzeugten Kamerabildern vorhanden sind. Dabei sollen Informationen, die für bestimmte Dienste relevant sind, in den Bildern erhalten bleiben. Mit SpAn sollen Informationen über das Fahrverhalten des Benutzers, die sich aus den Geschwindigkeitsdaten des Fahrzeuges ableiten lassen, geschützt werden. Die Konzepte werden im Verlauf dieser Arbeit erläutert und evaluiert.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Algorithmen

Abkürzungsverzeichnis

ADGAN Auto-Driving Generative Adversarial Network. 22

AF Autonomes Fahrzeug. 19, 20, 22

CaDaA Camera Data Anonymization. 3, 5, 7, 11, 18, 41, 42, 43, 44, 49

CC Connected Car. 3, 5, 15, 17, 19, 20, 21, 22, 25, 41, 42, 43, 49

DLS Dummy Location Selection. 21

Lo*k*A Location *k*-Anonymity. 3, 5, 11, 15, 17, 33, 34, 35, 36, 37, 38, 39, 49

PL Privacy-Level. 23, 24, 25, 28, 29, 30, 31, 49

PLEvS Privacy-Level Evaluation System. 3, 5, 7, 15, 17, 23, 24, 26, 28, 30, 49

QoS Quality of Service. 17, 18, 24

SDC Software-Defined-Car. 15, 17, 21, 41, 49

SitAC Situation-aware Access Control. 20

SpAn Speed Anonymization. 3, 5, 7, 11, 18, 45, 46, 47, 48, 49

1 Einleitung

In modernen Autos werden immer mehr Daten erfasst und verarbeitet. Sogenannte *Connected Cars* (CCs) können sich mit anderen Autos verbinden und Informationen austauschen, oder über das Internet mit bestimmten Diensten kommunizieren. Dieser Datenaustausch ist essentiell für zentrale Dienstleistungen eines CCs. Zum Beispiel können durch Verarbeitung der Sensor- und Kameradaten potentielle Gefahren erkannt und Unfälle verhindert werden. Dies gewährt eine erhöhte Sicherheit der Insassen eines solchen Fahrzeuges und anderer Verkehrsteilnehmer. Auch um autonomes Fahren zu ermöglichen sind diese Daten von großer Bedeutung.

Zu den erfassten Daten können zum Beispiel Ortsinformationen, Kameradaten, Informationen über die Geschwindigkeit des Fahrzeugs und weitere (Sensor-)Daten gehören. Selbstverständlich ist es für den Benutzer von zentraler Bedeutung, dass diese Daten sensibel behandelt werden und seine Privatsphäre ausreichend geschützt ist. Sämtliche Daten können Informationen enthalten, die für den Benutzer als schützenswert empfunden werden. In der Regel liegt es im Interesse eines Benutzers, dass seine Daten anonymisiert werden, bevor diese von dezentralen Diensten verarbeitet werden.

Die Schützenswürdigkeit dieser Daten ist häufig situationsabhängig. Befindet man sich zum Beispiel in der Nähe seines Zuhauses, enthalten Daten wie die Kamerabilder von solchen Fahrzeugen in der Regel sensiblere Informationen als wenn man aktuell zum Beispiel auf einer Autobahn fährt.

Diese Bachelorarbeit beschäftigt sich mit der situationsbezogenen Anonymisierung von Daten im Kontext von *Connected-* beziehungsweise *Software-Defined-Cars* (SDCs).

Die Arbeit ist in folgender Weise gegliedert: In Kapitel 2 wird die Problemstellung des Themas und die Ziele der Arbeit erläutert. Kapitel 3 stellt Arbeiten vor, die einen Bezug zum Thema oder den in dieser Arbeit vorgestellten Konzepten haben.

Darauffolgend werden die in der Arbeit ausgearbeiteten Konzepte vorgestellt. In Kapitel 4 wird PLEvS eingeführt, ein Entwurf zur Bestimmung von Anonymisierungsgraden anhand Auswertung modellierter Situationen. Kapitel 5 befasst sich mit Lo*k*A, einem Algorithmus der Ortsdaten mittels *k*-Anonymität anonymisiert. Darauf wird in Kapitel 6 ein Konzept zur Anonymisierung von Kameradaten vorgestellt. In Kapitel 7 folgt der Entwurf eines Algorithmus, mit dem Geschwindigkeitsdaten simuliert werden können. Zum Schluss werden in Kapitel 8 die Ergebnisse der Arbeit zusammengefasst und ein kurzer Ausblick gestellt.

2 Problemstellung und Ziele der Arbeit

Im Kontext von SDCs beziehungsweise CCs ist der Schutz der Privatsphäre von Benutzern ein Thema enormer Wichtigkeit. Unter den riesigen Datenmengen, die erzeugt werden, befinden sich jede Menge sensible Informationen über Insassen des Fahrzeuges und dessen Umgebung.

Dabei existiert häufig ein Trade-Off zwischen der *Quality of Service* (QoS) und dem Schutz der Privatsphäre. Etwa vervielfachen k-Anonymisierungs-Algorithmen, die "Dummy"-Elemente erzeugen, den Berechnungsaufwand für einen Dienstanbieter, da von $k - 1$ Anfragen "umsonst" verarbeitet werden, um die Anonymität des Benutzers zu schützen. Bei anderen Ansätzen, zum Beispiel der direkten Modifikation der Daten, kann es zu einer sinkenden QoS für den Benutzer kommen, da die Genauigkeit von Algorithmen, die auf den modifizierten Daten arbeiten, leiden kann.

Wie dieser Trade-Off sinnvoll festgelegt wird, ist häufig situationsbedingt. In bestimmten Situationen priorisiert ein beispielhafter Benutzer den Schutz seiner Privatsphäre, während in anderen Situationen die QoS als wichtiger erachtet wird. An Orten wie dem Wohnort des Benutzers liegt es in der Regel im Interesse der entsprechenden Person, dass bestimmte Daten sensibler behandelt werden als an anderen Orten. Dies könnten zum Beispiel Orts- beziehungsweise Kameradaten sein. Sollte sich das Auto in der Nähe des Wohnortes besitzen, sollten diese Daten stärker anonymisiert werden.

Es könnte allerdings auch eine Situation eintreffen, in welcher der Benutzer die QoS unabhängig seines aktuellen Ortes als deutlich wichtiger als seine Privatsphäre empfindet. Zum Beispiel im Falle eines Unfalls sollen Rettungskräfte alle relevanten Informationen über das Fahrzeug und dessen Insassen erhalten. In dieser Situation soll keine Anonymisierung auf die entsprechenden Daten angewandt werden.

Ein Thema, mit dem sich diese Bachelorarbeit auseinandersetzt, ist daher die Definition und Evaluation von Situationen. Hierbei ist die zentrale Problemstellung, wie man aus den (verteilt vorliegenden) Daten eines CCs Aussagen über den aktuellen Zustand des Fahrzeuges und der Umgebung festlegen und auswerten lassen kann. Je nach dem aktuellen Zustand des Fahrzeuges, ergo welche Situationen gerade eintreffen (und nicht eintreffen), sollen bestimmte Anonymisierungsmechanismen unterschiedlich stark greifen.

Mit PLEvS wird ein Konzept vorgestellt, mit dem sich (benutzerdefinierte) Situationen festlegen und auswerten lassen können. Abhängig vom aktuellen Stand der Situationen lässt sich definieren, wann und wie stark bestimmte Daten, die zu bestimmten Zielen gesendet werden sollen, anonymisiert werden sollen. Die Situationen werden kontinuierlich ausgewertet, um den aktuellen Zustand von Fahrzeug und Umgebung akkurat abzubilden.

Darüber hinaus werden Konzepte vorgestellt, die sich mit der konkreten Anonymisierung von bestimmten CC Daten auseinandersetzen. LokA ist ein auf k-Anonymität aufbauender Algorithmus, durch den Ortsdaten effektiv geschützt werden sollen. Der Algorithmus, der ein von Niu et al.

vorgestelltes Konzept erweitert [NLZ+14], zieht diverse Wahrscheinlichkeiten in Betracht, um die Identifizierung des realen Ortes zu verhindern, auch unter der Annahme, dass ein Dienstanbieter den Algorithmus und seine Parameter kennt.

CaDaA stellt einen Entwurf eines Algorithmus dar, mit dem Kameradaten anonymisiert werden, indem sensible Informationen aus ihnen entfernt werden. Dabei wird darauf geachtet, dass relevante Informationen wie die Straße oder andere Verkehrsteilnehmer auf den Bildern erhalten bleiben, um eine hohe QoS garantieren zu können. Situationsabhängig soll dabei festgelegt werden, welche Daten aus den Kamerabildern entfernt werden und welche erhalten bleiben.

Als Letztes befasst sich die Arbeit mit der Anonymisierung von Geschwindigkeitsdaten. Es könnte im Interesse eines Benutzers liegen, dass Informationen über das Fahrverhalten anonymisiert werden, bevor diese an Dienstanbieter wie zum Beispiel Versicherungsfirmen übermittelt werden. Dazu wird SpAn eingeführt, ein Konzept, mit dem Geschwindigkeitsverläufe simuliert werden sollen.

Alle in dieser Bachelorarbeit vorgestellten Konzepte wurden in Java beziehungsweise Python implementiert und ausgewertet.

3 Verwandte Arbeiten

Dieses Kapitel befasst sich mit verwandten Arbeiten, die einen direkten thematischen Bezug zu dieser Arbeit aufweisen oder Einflüsse auf die Konzepte der Arbeit haben. Die für diese Bachelorarbeit relevanten Themen und Problematiken, mit denen sich die entsprechenden Arbeiten auseinandersetzen, werden im Folgenden kurz erläutert.

Der erste Abschnitt beschäftigt sich mit Arbeiten, die sich generell mit dem Thema der Privatsphäre der (personenbezogenen) Daten eines CCs beziehungsweise *Autonomen Fahrzeuges* (AF) auseinandersetzen. Im darauffolgenden Abschnitt wird die Thematik aufgegriffen, wie Situationen modelliert und evaluiert werden können. Im dritten und vierten Abschnitt werden Arbeiten vorgestellt, die sich spezifisch mit der Anonymisierung von Orts- beziehungsweise Kameradaten beschäftigen.

3.1 Privatsphäre in Connected Car Szenarien

Viele Arbeiten greifen das Thema der Privatsphäre (und Sicherheit) bezüglich der von CCs beziehungsweise AFs erhobenen Daten auf.

Yankson befasst sich in seiner Arbeit mit den Bedenken und Risiken, die mit AFs und deren Daten verbunden sind [Yan20]. AFs erheben und verarbeiten eine große Menge an Daten, um sicheres autonomes Fahren zu ermöglichen. Ist kein ausreichender Schutz der Daten vorhanden, so können Angreifer nicht nur die Privatsphäre, sondern auch die Sicherheit eines AFs enorm gefährden.

Die Thematik wird ebenso von Dave et al. aufgegriffen [DSR19]. Sie erwähnen die Vorteile von AFs, zum Beispiel das verminderte Unfallrisiko und kleinere Reisezeiten durch Routenoptimierung einschließlich der damit verbundenen niedrigeren Belastung der Umwelt. Aber auch Dave et al. erkennen, dass mit der enormen Anzahl an verarbeiteten Daten mit Privatsphäre und Sicherheit verbundene Probleme entstehen. Hacker, die an die entsprechenden Daten gelangen, könnten an Informationen wie die Identität der Insassen, das Ort des Fahrzeuges oder ob der Fahrer zu einem bestimmten Zeitpunkt zu Hause ist, gelangen.

Wang et al. diskutieren über die Problematik, wie die Dienste eines AFs die Privatsphäre der Nutzer gefährden können, wenn diese nicht anonymisiert werden [WCY20]. Durch Anfragen von Benutzern, die Lokationsinformationen enthalten, können Dienstanbieter gegebenenfalls Rückschlüsse auf sensible Informationen der Person führen, wie zum Beispiel das Alter. Die Autoren nennen Suchanfragen (auf bestimmte Orte) als Beispiel. Aus diesen können Dienstanbieter sensible Informationen ableiten, zum Beispiel über die Religion einer Person, wenn eine Kirche gesucht wird, oder den Gesundheitszustand, falls der Benutzer ein Krankenhaus suchen sollte. Um dieser Problematik entgegenzuwirken, müssen entsprechende Anfragen effektiv anonymisiert werden, bevor sie gesendet werden.

Die Wichtigkeit des Schutzes der Privatsphäre von Benutzern ist auch aus rechtlicher Perspektive enorm. Akca et al. beschäftigen sich mit den legalen Aspekten des Themas [AKA20]. Hierbei beziehen Sie sich auf die in der EU am 25. Mai 2018 in Kraft getretene Datenschutz-Grundverordnung GDPR [Eur16]. Die Prinzipien der GDPR setzen voraus, dass personenbezogene Daten transparent, vertraulich und mit Integrität behandelt werden müssen.

Personenbezogene Daten, die in AFs beziehungsweise CCs gesammelt und verarbeitet werden, müssen also entsprechend geschützt werden. Dies ist nicht nur aus moralischer Sicht wichtig, sondern auch um potentielle Probleme mit dem Gesetz zu vermeiden. Werden die Daten nicht entsprechend sicher und sensibel behandelt, so kann dies das Vertrauen in entsprechende Autos und ihre Dienste erheblich senken.

3.2 Situationsmodellierung

Um ein situationsabhängiges System zum Schutz der Privatsphäre zu entwerfen, muss man sich mit der Frage beschäftigen, wie man Situationen modellieren und auswerten kann.

Mit dieser Thematik befassen sich Hüffmeyer. et al. in ihrer Arbeit [HHM+17]. Sie stellen *Situation-aware Access Control* (SitAC) vor, ein System, mit dem situationsabhängig Zugriffsanfragen auf sensible Daten erlaubt oder abgelehnt werden sollen. Hüffmeyer. et al. nennen betreutes Wohnen als einen der Anwendungsfälle von SitAC. Als Beispiel erläutern sie ein Szenario, in dem eine altere Person hinfällt, was durch (Bewegungs-)Sensoren detektiert wurde. In diesem Fall sollen Rettungsdienste für einen bestimmten Zeitraum Zugriff auf in der Wohnung installierte Kameras erhalten, um die Situation schnell einschätzen zu können. Familienmitglieder hingegen sollen jederzeit uneingeschränkten Zugriff auf die Kameradaten haben, während Zugriffsversuche von unautorisierten Angreifern in jedem Fall abgelehnt werden sollen. Die Situationen werden mittels eines *Situation Templates* modelliert. Dabei werden Sensordaten mit logischen Verknüpfungen (**AND**, **OR**, **XOR**) verbunden und ausgewertet. Geräte werden mit einem Besitzer definiert. Standardmäßig hat nur der Besitzer Zugriff auf das Gerät. Weitere Benutzer bzw. Anwendungen können registriert werden, sodass diese bei Eintritt von bestimmten Situationen benachrichtigt werden.

Soll ein Benutzer, der standardmäßig keinen Zugriff auf ein Gerät hat, in einer bestimmten Situation Zugriff erhalten, so erhält er diesen maximal für einen bestimmten Zeitintervall. Situationseinträge bestehen aus einer *id*, einem *occurred* Feld, welches angibt, ob die Situation aktuell aktiv ist, einem Zeitstempel *time*, der angibt, wann die Situation eingetreten ist, und einem *accessInterval*, der festlegt, wie lange der Zugriff maximal gewährt werden soll. Eine Zugriffsanfrage enthält einen Zeitstempel *time*, der angibt, zu welchem Zeitpunkt die Anfrage gestellt wurde. Wenn die Situation nicht aktiv ist, oder der Zeitpunkt der Anfrage nicht zwischen der Eintrittszeit der Situation und dem Ablauf des Zugriffsintervalls ist, wird die Anfrage abgelehnt.

Auf diese Art und Weise kann das System so eingerichtet werden, dass bestimmte Nutzer nur in bestimmten Situationen für vorbestimmte Zeiträume Zugriff auf definierte Geräte erhalten.

3.3 Anonymisierung von Ortsdaten

Niu et al. diskutieren in ihrer Arbeit die Problematiken der Anonymisierung von Ortsdaten im Kontext von lokationsbasierten Diensten [NLZ+14]. Die Autoren befassen sich mit k-Anonymisierungs-Algorithmen, welche die Privatsphäre der Nutzer schützen sollen, und den Problemen von existierenden Ansätzen. Als eine der Problematiken nennen sie die Unausgewogenheit der Wahrscheinlichkeiten, dass eine Anfrage von einem bestimmten Ort gestellt wird. Erzeugt der k-Anonymisierung-Algorithmus "Dummy"-Orte, die offensichtlich nicht der echte Ort sind, so ist es für den Dienstanbieter nicht schwer den echten Ort der Benutzers mit hoher Wahrscheinlichkeit zu identifizieren.

Ein Beispiel zur Veranschaulichung: Ein Benutzer, im Kontext dieser Arbeit ein Fußgänger, möchte auf einem begehbarem Weg einen lokationsbasierten Dienst (privatisiert) in Anspruch nehmen. Dazu soll eine k-Anonymisierung mit $k = 5$ stattfinden, also müssen vier Dummy-Orte erzeugt werden. Wenn der Algorithmus nun schwer begehbare Orte als Dummies auswählt, zum Beispiel in einem Fluss oder einem Sperrgebiet, dann ist es nicht schwer Rückschlüsse auf den echten Ort des Benutzers zu führen. Der Dienstanbieter könnte in einem solchen Fall also leicht den echten Standort des Benutzers herausfinden.

In ihrer Arbeit stellen Niu et al. *Dummy Location Selection* (DLS) vor, ein Algorithmus der k-Anonymität erzeugt und der genannten Problematik entgegenwirkt. Dazu wird angenommen, dass die Lokationskarte als ein $n \times n$ Gitter gleichgroßer Zellen vorliegt. Jeder Zelle wird eine Wahrscheinlichkeit zugeordnet, abhängig davon wie viele Anfragen in der Vergangenheit von einem in der Zelle liegendem Ort gestellt wurden. Durch Kombination von Zufälligkeit und der Entropiefunktion soll eine Menge ausgewählt werden, die neben dem realen Ort $k - 1$ Dummy-Orte enthält, während die Wahrscheinlichkeiten aller Orte möglichst nah beieinander liegen.

Die Entropiefunktion ist gegeben durch

$$H = -\sum_{i=1}^{k} p_i \cdot \log_2 p_i$$

Eine höhere Entropie H bedeutet, dass es schwerer ist, den echten Ort unter den Dummy-Orten zu identifizieren. Ziel ist es also, die Entropie möglichst zu maximieren. Die Funktionsweise von DLS ist wie folgt:

Zunächst werden die n^2 Zellen in einer Liste nach ihren Wahrscheinlichkeiten sortiert. Haben mehrere Elemente die gleiche Wahrscheinlichkeit wie die echte Zelle, werden diese so sortiert dass die Hälfte dieser Elemente vor und die andere Hälfte nach der echten Zelle auftreten. Aus dieser Liste werden nun $2k$ Elemente entnommen: Jeweils die k Elemente vor und nach der echten Zelle. Mit diesen $2k$ Elementen werden nun m Mengen gebildet, die $k - 1$ zufällig gewählte Zellen und die echte Zelle enthalten. Hierbei ist m ein wählbarer Parameter. Anschließend werden die Entropien H_j mit den Wahrscheinlichkeiten der Elemente von jeder der erzeugten Mengen berechnet, und diejenige bestimmt, deren Elemente untereinander die größte Entropie besitzen. Diese Menge wird schlussendlich gewählt, um die Anfrage zu anonymisieren.

Mit DLS stellen Niu et al. einen Ansatz zur Optimierung von k-Anonymitäts-Algorithmen vor, bei dem Rückschlüsse auf den echten Ort des anfragenden Nutzers minimiert werden sollen. Der Ansatz von Niu et al. lässt sich auch auf CC bzw. SDC Szenarien übertragen, mit denen sich diese Bachelorarbeit befasst.

3.4 Anonymisierung von Kameradaten

Es existieren viele Arbeiten, die sich mit der Anonymisierung von Kameradaten beschäftigen.

Mit der Thematik des Erhalts der Privatsphäre bei Kamerabildern, die von einem AF erzeugt und verarbeitet werden, beschäftigen sich Xiong et al. [XLHC19]. Die Autoren diskutieren, wie diese Daten in die Privatsphäre von Nutzern beeinflussen. Sie behaupten, dass Angreifer, die Kameradaten eines AFs gestohlen haben, Rückschlüsse auf den Ort des Fahrzeuges führen könnten, um daraus weitere Informationen abzuleiten. Zum Beispiel kann sich an Gebäuden, die sich im Hintergrund der Kamerabilder befinden, der aktuelle Aufenthaltsort des Fahrzeuges bestimmen lassen. Diese Hintergrundinformationen sind für die Funktionen eines AFs zum Großteil überflüssig.

Um dieses Problem zu lösen stellen Xiong et al. *Auto-Driving Generative Adversarial Network* (ADGAN) vor, ein GAN-basierter Ansatz zur Entfernung von Hintergrundinformationen in Kameradaten von AFs. Der Algorithmus soll Hintergrunddaten, die Rückschlüsse auf den Ort des Fahrzeuges erlauben, aus den Kamerabildern entfernen. Mit ADGAN soll die Privatsphäre von Nutzern geschützt werden ohne die Funktionalitäten eines AFs zu beeinträchtigen. Laut Xiong et al. ist ihre Arbeit (zum Veröffentlichungszeitpunkt) die Erste, die sich mit den Problemen der Privatsphäre der Kameradaten eines AFs beschäftigt.

Es gibt allerdings weitere Arbeiten die sich mit der Anonymisierung von Bilddaten in anderen Kontexten auseinandersetzen, deren Prinzipien sich aber auch auf die CC Thematik übertragen lassen.

Yu et al. entwarfen *iPrivacy*, ein Tool, das automatisch sensible Daten in (von Personen) aufgenommenen Bildern erkennen und entfernen soll [YZK+17]. Nutzer können festlegen, welche Objekte aus den Bildern entfernt werden sollen. Mittels eines auf maschinellen Lernen basierendem Algorithmus werden Bilder in Klassen segmentiert, und entsprechende Objekte aus dem Bild entfernt oder unkenntlich gemacht.

Eine weitere Arbeit, die sich allgemeiner mit Bildsegmentierung beschäftigt, stammt von Porzi et al. [PBCK19]. Die Autoren stellen eine ebenso auf maschinellem Lernen basierten Architektur vor, um Objekte auf (Straßen-)Bildern zu erkennen und die Bilder entsprechend zu segmentieren.

In dieser Bachelorarbeit soll die Thematik der Anonymisierung von Kameradaten zum Schutz der Privatsphäre (in einem CC-Kontext) ebenso aufgegriffen werden.

4 PLEvS - Evaluation von Situationen

Das *Privacy-Level Evaluation System* (PLEvS) stellt ein Konzept dar, um Daten situationsabhängig mit verschiedener Stärke zu anonymisieren.

PLEvS besteht aus drei Komponenten: Einem oder mehreren *Verarbeitungssystemen*, in denen die zu anonymisierenden Daten verarbeitet werden, dem eigentlichen *Evaluationssystem*, in dem die anzuwendenden *Privacy-Levels* (PLs) durch Auswertung von Situationen bestimmt werden, und einem oder mehreren *Attributssystemen*, welche Input-Daten zu definierten Attributen verarbeiten. Die im Evaluationssystem festgelegten Situationen sind abhängig von den definierten Attributen. Ob eine Situation eintrifft oder nicht lässt sich über die Werte der Attribute bestimmen. Die genauen Funktionsweisen der Komponenten werden in den folgenden Abschnitten dieses Kapitels erläutert.

Die Komponenten kommunizieren nach dem *Publish-Subscribe* Prinzip miteinander. Die Verarbeitungssysteme abonnieren für sie relevante PL Updates, um vom Evaluationssystem über entsprechende Änderungen informiert zu werden. Das Evaluationssystem abonniert Updates auf die Werte der definierten Attribute, um Situationen abhängig vom aktuellen Zustand des Fahrzeuges evaluieren zu können.

Dazu wird ein *Main-Topic* definiert, welches in allen Komponenten identisch sein muss. In denen in diesem Kapitel gezeigten Beispielen wird `plevs` standardmäßig als das Main-Topic festgelegt.

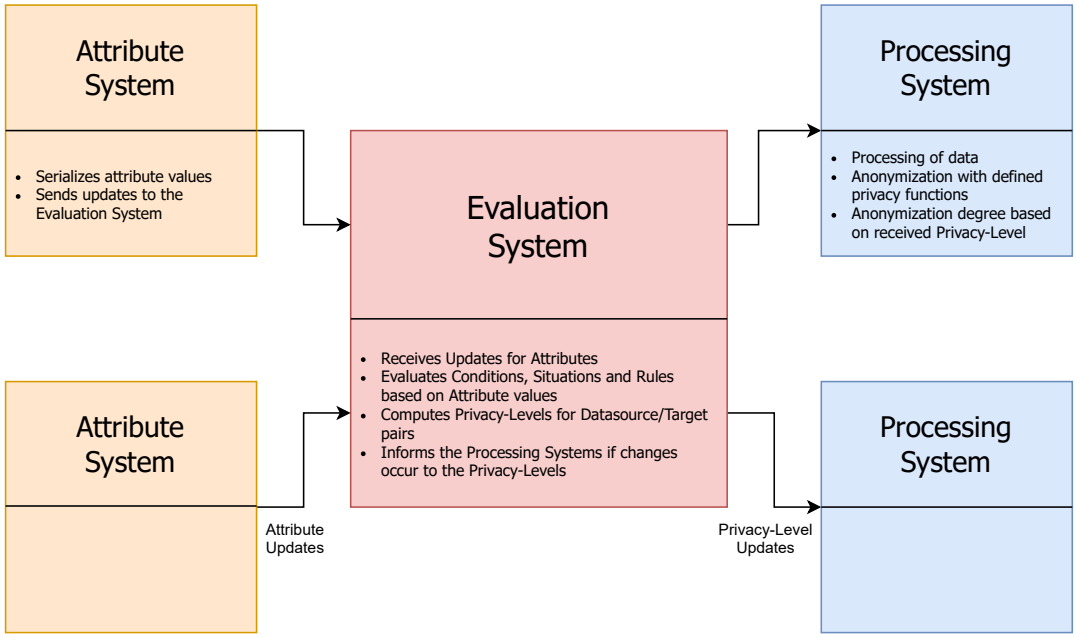


Abbildung 4.1: Grundstruktur von PLEvS

4.1 Verarbeitungssystem

Ein Verarbeitungssystem besteht aus mindestens einem *Verarbeitungs-Kanal*. In einem Verarbeitungs-Kanal werden Daten abhängig von seinem aktuellen PL anonymisiert.

Jeder Kanal besitzt die folgenden Informationen: Eine ID der *Datenquelle*, die angibt, welche Datenart in dem jeweiligen Kanal verarbeitet wird, die ID von einem *Datenziel*, das die entsprechenden Daten erhalten soll (z. B. ein Dienstanbieter), eine Liste von PLs, die jeweils eine anzuwendende Anonymisierungsfunktion enthalten, aufsteigend nach dem Anonymisierungsgrad der Funktion sortiert, und das aktuell angewendete PL.

Ein Privacy-Level besteht aus einer Anonymisierungs-Funktion inklusive festgelegter Parameter der Funktion. Das PL 0, also das PL, das den niedrigsten Anonymisierungsgrad aufweisen soll, ist bei jedem Kanal mit der Identitätsfunktion *id* festgelegt. Auf diesem Level werden Daten unanonymisiert weitergegeben. Höhere PLs lassen sich beliebig definieren.

Ein Beispiel: Es sollen Anfragen an einen lokationsbasierten Dienst `LBS` situationsabhängig verschieden stark anonymisiert werden. Dazu definieren wir einen Kanal, in dem wir `locationData` als ID für die Datenquelle festlegen. Die Datenziel-ID wird als `LBS` definiert. Zudem soll zwischen drei PLs unterschieden werden, auf denen die Ortsdaten verschieden stark anonymisiert werden. Dies soll mit einer k-Anonymisierungs-Funktion, die Dummy-Orte zu dem echten Ort auswählt, geschehen. Auf dem vordefinierten PL 0 werden die Anfragen nicht anonymisiert. PL 1 wird mit einer entsprechenden k-Anonymisierungs-Funktion definiert, mit Parameter $k = 4$. Level 2 wird mit derselben Funktion definiert, mit Parameter $k = 10$.

Die PLs sollen situationsabhängig in Kraft treten. In bestimmten Situationen soll PL 0 in Kraft treten, in anderen PL 1, und in wiederum anderen PL 2. Dadurch steigt der Anonymisierungsgrad auf Kosten der QoS wegen des höheren Berechnungsaufwandes. Die Definition und Evaluation von Situationen findet im Evaluationssystem statt. Dies wird später im Abschnitt 4.2 genauer erläutert.

Damit die entsprechenden Kanäle PL Updates vom Evaluationssystem erhalten, werden die relevanten Topics für alle Anfragetypen abonniert. Ein *Anfragetyp* ist definiert als ein Paar, bestehend aus einer Datenquelle-ID und einer Datenziel-ID.

```
{Main-Topic}/privacyLevels/{Datenquelle-ID}/{Datenziel-ID}
```

Im Beispiel wird also das Topic

```
plevs/privacyLevels/locationData/LBD
```

vom Verarbeitungssystem abonniert. Dadurch wird der entsprechende Kanal benachrichtigt, falls sich sein PL ändern soll, weil sich der Status einer relevanten Situation geändert hat.

4.2 Evaluationssystem

Im Evaluationssystem werden die PLs bestimmt, die in den Kanälen der Verarbeitungssysteme anzuwenden sind. Dazu werden voneinander abhängige *Attribute*, *Konditionen*, *Situationen* und *Regeln* definiert. Durch Evaluation der Situationen wird bestimmt, welche Regeln für welche Anfragetypen angewendet werden. Daraus werden die PLs abgeleitet und Änderungen auf den entsprechenden Topics publiziert.

4.2.1 Datenquellen und Datenziele

Zunächst werden im Evaluationssystem die IDs von allen Datenquellen und Datenzielen registriert, wie sie in den Verarbeitungssystemen vorliegen. Wie bereits erläutert, bildet ein Paar aus der ID einer Datenquelle und der ID eines Datenzieles einen *Anfragetyp*. Für jeden Anfragetyp wird ein PL bestimmt.

Bei Änderungen der PLs werden die Verarbeitungssysteme über die jeweiligen Topics benachrichtigt, wodurch sich die im Verarbeitungssystem angewandten Anonymisierungsfunktionen ändern. Zudem wird für jede Datenquelle ein *Standard Privacy-Level* definiert. Standardmäßig besitzen alle Anfragetypen das festgelegte Standard PL der entsprechenden Datenquelle.

4.2.2 Attribute

Durch Attribute erhält das Evaluationssystem Informationen über den aktuellen Zustand des Fahrzeuges und der Umgebung. Es lassen sich beliebig viele Attribute definieren, die verschiedene Werte eines CC-Szenarios repräsentieren sollen. Ein Attribut besteht aus einer (eindeutigen) *ID*, einem festgelegten Datentyp *type* und einem Wert *value*, der dem Datentyp entspricht. Die Werte der Attribute werden durch Benachrichtigungen vom Attribut-System aktualisiert. Dazu werden vom Evaluationssystem die relevanten Topics für jedes definierte Attribut abonniert:

```
{Main-Topic}/attributes/{Attribut ID}
```

Besitzt das Evaluationssystem zum Beispiel das Attribut `speed`, welches die aktuelle Geschwindigkeit des Fahrzeuges repräsentieren soll, wird das Topic

```
plevs/attributes/speed
```

abonniert. Dadurch erhält das Evaluationssystem relevante Attribut-Updates, und die entsprechenden Werte werden in den Attribut-Objekten aktualisiert.

4.2.3 Konditionen

Die Auswertung der Attribute erfolgt mittels Konditionen. Mit Konditionen lassen sich simple oder komplexe Bedingungen definieren, die abhängig von den aktuellen Werten der Attribute wahr oder falsch sind. Bei Zustandsveränderungen können sich die Resultate der einzelnen Konditionen ändern.

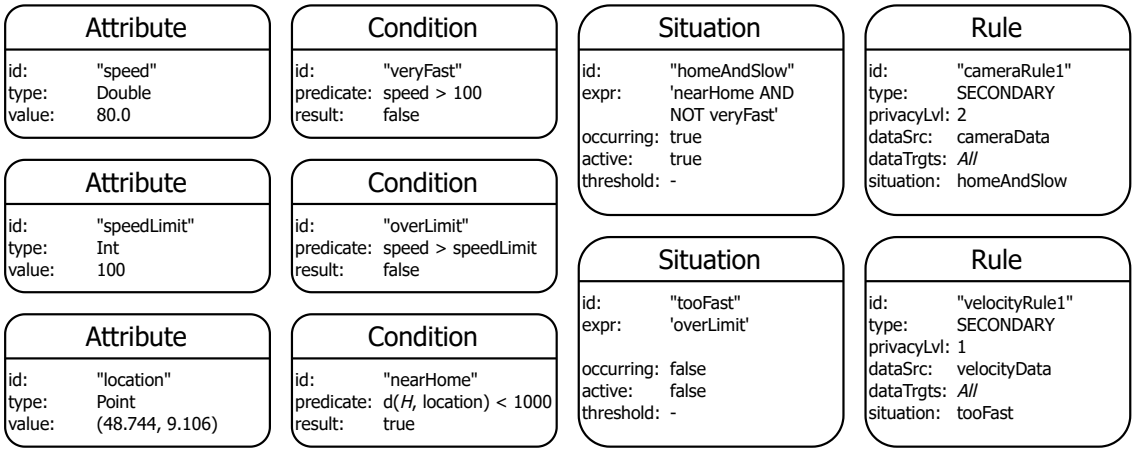


Abbildung 4.2: Beispielobjekte eines Evaluationssystems

Jede Kondition besitzt eine *ID*, eine Prädikatsfunktion *predicate*, die ein oder zwei Attribute als Parameter enthält und abhängig von dessen Werten einen booleschen Wert erzeugt, und einen Wahrheitswert *result*, welcher das Resultat dieser Funktion speichert. Wenn sich der Wert eines abhängigen Attributes einer Kondition ändert, so wird diese informiert und die entsprechende Prädikatsfunktion neu evaluiert. Abhängig davon ändert sich gegebenenfalls das Resultat der Kondition.

Als Beispiel wird eine Kondition `nearHome` definiert, die genau dann wahr werden soll, wenn sich das Fahrzeug näher als 1000 Meter zu einem bestimmten Punkt H befindet. Die Konstante H ist in diesem Beispiel als der Punkt, an dem sich das Zuhause des Benutzers befindet, festgelegt.

In diesem Fall wird

$$pred = d(H, location) < 1000$$

als Prädikatsfunktion der Kondition gewählt, wobei d eine Funktion ist, die die Distanz zwischen zwei (geographischen) Punkten in Meter berechnet, und `location` der Wert eines entsprechenden Attributs, welches den aktuellen Ort des Fahrzeuges beschreibt. (vgl. Abbildung 4.2).

Immer wenn der Wert des Attributes `location` aktualisiert wird, wird die Kondition `nearHome` neu evaluiert. Unterscheidet sich das Resultat der Auswertung vom Vorherigen, wird der Wert entsprechend aktualisiert.

4.2.4 Situationen

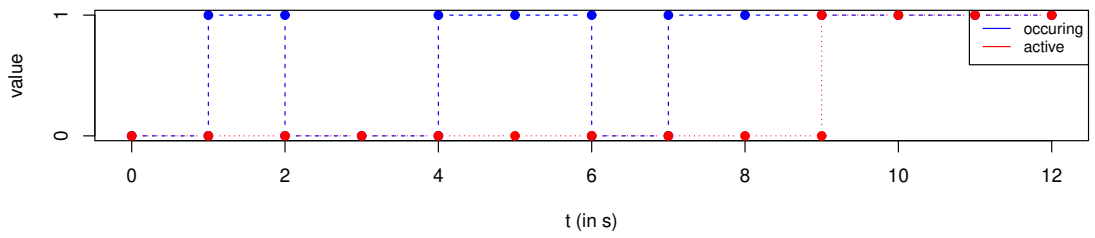
Aus den Konditionen lassen sich Situationen definieren und evaluieren. Jede Situation besitzt neben einer *ID* zwei Wahrheitswerte: Einen *occurring* Wert, der angibt, ob die Situation aktuell eintrifft, und einen *active* Wert, der bestimmt, ob die Situation im Moment aktiv ist. Zwischen den beiden Begriffen wird bewusst unterschieden. Die Bedingungen, ab wann eine eintreffende Situation aktiv wird und umgekehrt, werden durch eine optionale *Schwelle* festgelegt. Diese werden im folgenden Unterabschnitt genauer beschrieben. Ist keine Schwelle definiert, gilt zu jedem Zeitpunkt $active = occurring$.

Ob eine Situation eintrifft oder nicht, wird durch einen *Ausdruck* ermittelt. Ein Ausdruck besteht aus einer oder mehreren Konditionen, die mit den logischen Operatoren **AND**, **OR**, **XOR** und **NOT** verknüpft werden. Aus den Resultaten der Konditionen lässt sich das Ergebnis des Ausdrucks berechnen. Ändert sich das Resultat einer abhängigen Kondition, wird das Situations-Objekt informiert und der Ausdruck neu ausgewertet. Dadurch ändert sich gegebenenfalls der Wert der *occurring* Variable.

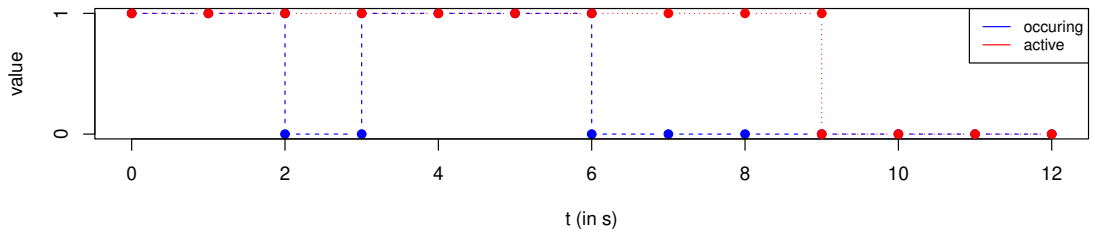
Schwellen

Wenn sich der Wert eines Attributes an einem Schwellwert einer Kondition befindet und häufig aktualisiert wird, kann es vorkommen, dass sich das Resultat der Kondition sehr häufig in einem kurzen Zeitraum ändert. Dies kann entsprechende Auswirkungen auf Situationen haben, die von dieser Kondition abhängig sind. Um diesem Verhalten entgegenzuwirken, wird die *Schwelle* eingeführt. Mit Schwellen soll verhindert werden, dass der Status einer Situation sehr häufig in einem kleinen Zeitraum umschaltet.

Eine Schwelle besteht aus einer (positiven) Zeitdauer *duration*, einem *activationThreshold* und einem entsprechenden *deactivationThreshold*. Dabei muss für die beiden Schwellwerte die Bedingung $0 \leq deactivationThreshold < activationThreshold \leq 1$ erfüllt sein.



(a) Aktivierungsverlauf einer Situation mit Schwelle



(b) Deaktivierungsverlauf einer Situation mit Schwelle

Abbildung 4.3: Auswirkungen von Schwellen auf den Aktiv-Status von Situationen

Durch die beiden Schwellwerte wird bestimmt, wann eine eintreffende Situation aktiv wird beziehungsweise eine nicht eintreffende Situation inaktiv. Eine inaktive Situation wird zu einem Zeitpunkt t aktiv, wenn im Zeitintervall $[t - duration; t]$ die Situation zu mindestens *activationThreshold* der Zeit eingetroffen ist. Analog dazu wird eine aktive Situation erst wieder inaktiv, wenn im entsprechenden Zeitintervall die Situation maximal zu *deactivationThreshold* der Zeit eingetroffen ist.

Ein Beispiel: Es wird für eine Situation eine Schwelle definiert mit den Werten $duration = 5000ms$, $activationThreshold = 0.8$ und $deactivationThreshold = 0.6$. Die entsprechende Situation wird nur dann aktiv, wenn sie in den letzten 5 Sekunden zu mindestens 80% der Zeit eingetroffen ist, also mindestens 4 Sekunden. Umgekehrt wird die Situation erst dann wieder inaktiv, wenn sie in 5 aufeinanderfolgenden Sekunden zu maximal 60% der Zeit eingetroffen ist, was 3 Sekunden entspricht.

In Abbildung 4.3 **(a)** wird ein beispielhafter Aktivierungsverlauf dargestellt. Zum Zeitpunkt $t = 0s$ ist die Situation inaktiv und trifft nicht ein. Es sei angenommen, dass dies ebenso für alle vorherigen Zeitpunkte $t < 0s$ gelte. Die Aktivierungsbedingung ist erstmals bei $t_0 = 9s$ erfüllt, da die Situation im Zeitintervall $[4s; 9s]$ für insgesamt 4 Sekunden eingetroffen war. Zu diesem Zeitpunkt wird die Situation auf aktiv geschalten.

Ein weiteres Beispiel wird durch Abbildung 4.3 **(b)** gegeben, die einen typischer Deaktivierungsverlauf zeigt. Die Situation sei zu allen Zeitpunkten $t \leq 0s$ eingetroffen und aktiv. Zum Zeitpunkt $t_0 = 9s$ ist die Deaktivierungsbedingung erfüllt, wodurch die Situation auf inaktiv umschaltet.

4.2.5 Regeln

Die Evaluation der PLs erfolgt durch Regeln. Durch Regeln wird für jeden definierten Anfragetyp ein PL bestimmt. Zur Erinnerung: Ein *Anfragetyp* ist definiert als ein Tupel, bestehend aus der ID einer Datenquelle und der ID eines Datenzieles.

Eine Regel besteht aus einer *ID*, einem Wert *dataSource*, welcher die ID der Datenquelle angibt, für welche die Regel angewendet werden kann, und einer Zahl *targetPrivacyLevel*, durch welche das anzuwendende PL festgelegt wird, falls die Regel angewendet wird. Optional können eine Situation und eine Liste an Datenzielen definiert werden. Eine Regel ist genau dann *aktiv*, wenn die zugeordnete Situation aktiv ist. Ist keine Situation definiert bleibt die Regel dauerhaft aktiv. Die Datenziel-Liste legt fest, für welche Datenziele die Regel in Kraft treten kann. Wird keine Liste angegeben, kann die Regel für jedes Datenziel angewendet werden.

Die Regeln werden in einer bestimmten Reihenfolge evaluiert. Die erste Regel, die angewendet wird, bestimmt das PL des Anfragetyps. Eine Regel kann genau dann für einen Anfragetyp angewendet werden, wenn

- die Regel *aktiv* ist,
- die ID der *Datenquelle* der Regel und des Anfragetyps gleich sind und,
- falls eine Liste an Datenzielen definiert ist, die ID des *Datenzieles* des Anfragetyps in der Liste enthalten ist.

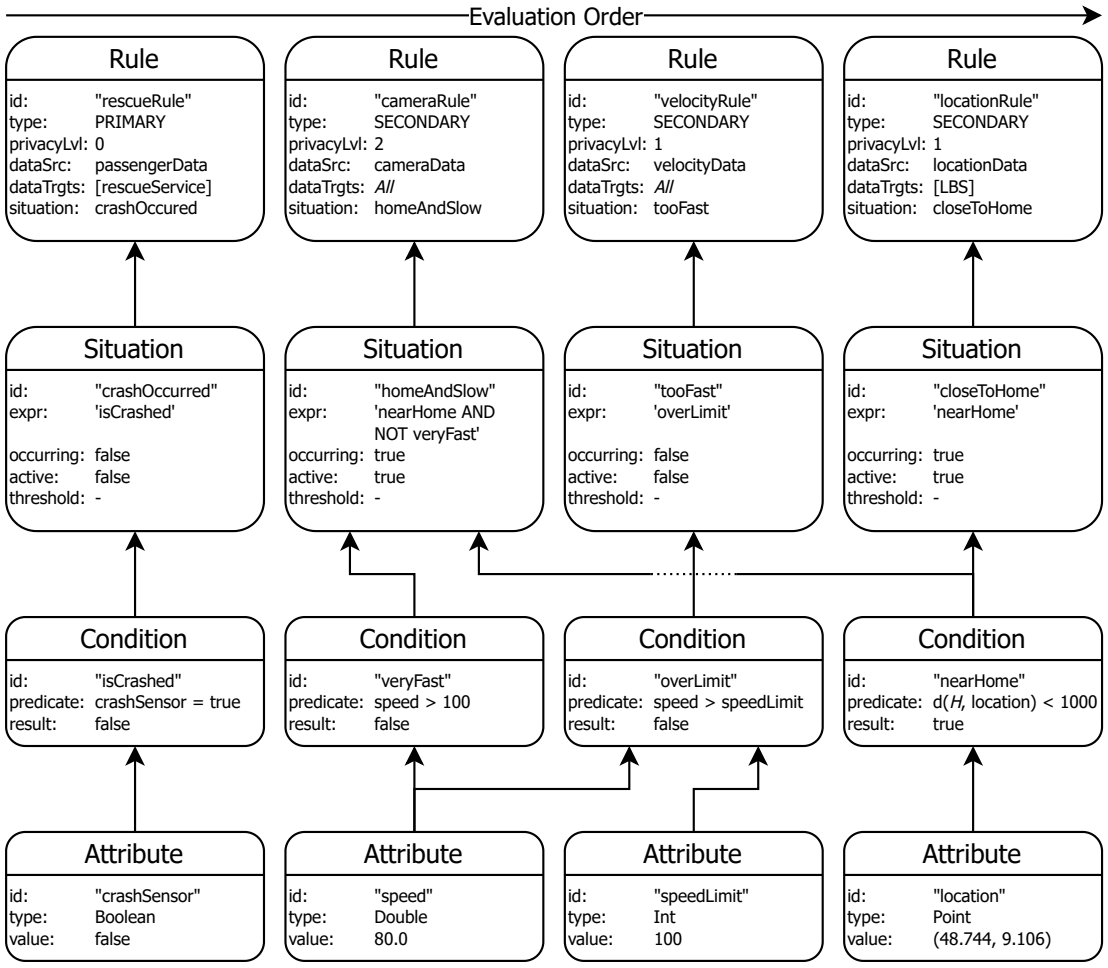


Abbildung 4.4: Beispielhafte Definition eines Evaluationssystems

Es wird zwischen *Primär-* und *Sekundärregeln* unterschieden. Primärregeln werden mit einer festgelegten Ordnung definiert, durch welche die Prioritäten dieser bestimmt werden. Die Primärregel mit der höchsten Priorität wird als erstes ausgewertet. Falls keine Primärregel angewendet werden kann, werden die Sekundärregeln evaluiert. Das maximale PL aller Sekundärregeln, die angewendet werden können, bestimmt dann das PL des Anfragetyps. Dazu werden die Sekundärregeln absteigend nach ihrem PL sortiert und in dieser Reihenfolge ausgewertet. Kann weder eine Primär-, noch eine Sekundärregel angewendet werden, wird das Standard-PL der Datenquelle für den Anfragetyp herangezogen.

Zusammenfassend werden die folgenen Schritte für jeden Anfragetyp durchgeführt, um das entsprechende PL zu ermitteln:

- Die Primärregeln werden in der festgelegten Ordnung ausgewertet. Das PL der ersten Regel, die angewendet werden kann, bestimmt das PL des Anfragetyps.
- Falls keine Primärregel zutrifft, werden die Sekundärregeln absteigend nach ihrem PL ausgewertet. Kann eine Sekundärregel angewendet werden, legt diese das PL des Anfragetyps fest.

- Kann keine Regel angewendet werden, wird das PL des Anfragetyps auf das Standard-PL der Datenquelle gesetzt.

Ändert eine Regel ihren Aktiv-Status, weil eine abhängige Situation aktiv oder inaktiv wurde, werden die PLs betroffener Anfragetypen neu evaluiert. Werden Änderungen festgestellt, werden diese auf den entsprechenden Topics publiziert, um die jeweiligen Verarbeitungssysteme zu informieren.

4.2.6 Beispiel

Eine beispielhafte Definition der Objekte eines Evaluationssystems ist in Abbildung 4.4 dargestellt. Das System besitzt die folgenden Attribute:

Attribut ID	Typ	Beschreibung
crashSensor	Boolean	Gibt an, ob der Crash-Sensor einen Unfall detektiert hat.
speed	Double	Stellt die aktuelle Geschwindigkeit des Fahrzeuges dar.
speedLimit	Int	Beschreibt das Tempolimit am aktuellen Ort.
location	Point	Die aktuellen Koordinaten des Fahrzeuges.

Tabelle 4.1: Attribute

Die Attribute werden laufend aktualisiert, um den aktuellen Zustand des Fahrzeuges zu repräsentieren. Eine beispielhafte Belegung der Attribute und die daraus resultierenden Werte der Konditionen und Situationen ist in der Abbildung gegeben.

Die Datenquellen werden mit folgenden Standard-PLs definiert:

Datenquelle-ID	Standard-PL	Beschreibung
passengerData	1	Informationen über Insassen.
cameraData	1	Aufgezeichnete Kameradaten.
velocityData	0	Daten über die Geschwindigkeit des Fahrzeuges.
locationData	0	Informationen über die aktuelle Position.

Tabelle 4.2: Datenquellen

Zur Veranschaulichung soll das PL des Anfragetyps `locationData/LBS` ermittelt werden. Dazu werden die Regeln in der dargestellten Reihenfolge ausgewertet.

Die Primärregel `rescueRule` kann nicht angewendet werden, da sie weder aktiv ist, noch Datenquelle und Datenziel übereinstimmen. Die Sekundärregel `cameraRule` ist zwar aktiv und kann für alle Datenziele angewendet werden, allerdings ist die Datenquelle eine andere. Dementsprechend kommt auch die Regel `velocityRule` nicht in Frage. Die erste Regel, die angewendet werden kann, ist die aktive Sekundärregel `locationRule`. Durch diese wird das PL des Anfragetyps bestimmt, welches in diesem Fall 1 ist.

Ändert sich der Aktiv-Status der Situation `closeToHome`, muss das PL des o. g. Anfragetyps neu evaluiert werden. In diesem Fall kann die entsprechende Regel nicht mehr angewendet werden. Da dies bedeutet, dass in diesem Fall keine Regel angewendet werden kann, wird das Standard PL der Datenquelle verwendet, in diesem Fall 0.

4.3 Attributssystem

Die Attributssysteme sind dafür zuständig kontinuierlich Input-Daten zu den Attribut-Werten aufzunehmen, zu serialisieren und auf den entsprechenden Topics zu publizieren. In der Implementierung werden die Objekte in das JSON-Format überführt.

Um den Wert eines Attributes zu aktualisieren, stellt das Attributssystem eine *update* Funktion bereit, die eine Attributs-ID und einen Wert vom entsprechenden Datentyp erhält. Soll beispielsweise das Attribut `location` mit einem neuen Wert aktualisiert werden, wird die Funktion mit der entsprechenden ID *"location"* und einem *Point* Objekt, welches die aktuellen Koordinaten des Fahrzeuges repräsentiert, aufgerufen.

Das Objekt wird anschließend auf dem Topic

```
plevs/attributes/location
```

in seiner serialisierten Form publiziert. Auf dem Evaluationssystem wird der Wert wieder deserialisiert und dem entsprechenden Attribut-Objekt zugeordnet. Durch eine kontinuierliche Aktualisierung der Werte kann das Evaluationssystem die Situationen abhängig vom aktuellen Zustand des Fahrzeuges evaluieren und die PLs auswerten.

```
{
  "x": 48.0,
  "y": 9.0
}
```

Listing 4.1: JSON Repräsentation eines Java Point2D Objektes

5 LokA - Anonymisierung von Ortsdaten

Damit ein k-Anonymisierungs-Algorithmus effektiv ist, ist es nötig, sich mit den Wahrscheinlichkeiten der zu anonymisierenden Elemente zu beschäftigen. Wie bereits von Niu et al. erläutert, sollten die von einem solchen Algorithmus gewählten Dummy-Elemente nicht im Bezug auf ihre Wahrscheinlichkeiten offensichtlich sein [NLZ+14].

Es ist allerdings auch von enormer Wichtigkeit, dass sich aus dem Algorithmus selbst nicht ableiten lassen kann, welches Element das reale Element ist. Dies könnte zum Beispiel durch Auswertung der Wahrscheinlichkeiten erfolgen, dass bestimmte Dummy-Elemente zu bestimmten echten Elementen erzeugt werden. Nimmt man bei k gegebenen Elementen an, dass ein Element das Echte ist, kann man, wenn man die Parameter des Algorithmus kennt, die Wahrscheinlichkeit berechnen, dass unter dieser Annahme genau diese Dummy-Elemente gewählt wurden. Analysiert man alle k Elemente auf diese Weise, können sich Rückschlüsse ergeben, falls es starke Unterschiede zwischen den entsprechenden Wahrscheinlichkeiten gibt.

Ein einfaches Beispiel: Um Ortsdaten zu anonymisieren wird ein solcher Algorithmus mit $k = 2$ parametrisiert. Dieser soll neben dem echten Ort einen Dummy-Ort bestimmen, bevor eine Anfrage an einen Dienstanbieter gestellt wird. Der Dienstanbieter erhält zwei Anfragen mit zwei unterschiedlichen Orten $\{a, b\}$, ohne zu wissen, welches der reale Ort und welches der Dummy-Ort ist. Kennt der Dienstanbieter den Algorithmus (und Parameter), so kann dieser die bedingten Wahrscheinlichkeiten $P(\{a, b\} \mid r' = a)$ und $P(\{a, b\} \mid r' = b)$ berechnen, die angeben, wie wahrscheinlich es ist, dass der Algorithmus die Menge $\{a, b\}$ erzeugt, unter der Annahme, dass a beziehungsweise b der echte Ort ist.

Sei angenommen, dass sich die Werte

$$P(\{a, b\} \mid r' = a) = 10\%$$

$$P(\{a, b\} \mid r' = b) = 0.01\%$$

ergeben. Unter der Annahme, dass a der reale Ort r' ist, wurde b mit einer Wahrscheinlichkeit von 10% als Dummy-Ort gewählt. Geht man hingegen davon aus, dass b der echte Ort ist, wurde a lediglich zu 0.01% als Dummy-Ort gewählt. Der Dienstanbieter kann daraus mit hoher Sicherheit schließen, dass in diesem Fall a der echte Ort des Benutzers ist. Wäre b der echte Ort, dann wäre es deutlich wahrscheinlicher, dass sich ein anderer Dummy-Ort als a ergeben hätte.

In diesem Kapitel wird *Location k-Anonymity* (LokA) vorgestellt, ein Konzept eines k-Anonymisierungs-Algorithmus, der Dummy-Orte sowohl nach ihren allgemeinen Wahrscheinlichkeiten, als auch den o.g. bedingten Wahrscheinlichkeiten auswählt. Dadurch sollen Rückschlüsse auf den echten Ort des Benutzers, die sich durch Analyse des Algorithmus ergeben können, minimiert werden.

5.1 Konzept

Gegeben sei eine Liste L, die n Objekte enthält, in diesem Fall Orte. Alle Orte besitzen einen sog. *Häufigkeits-Wert frequency*, der angibt, wie oft der entsprechende Ort in der Vergangenheit für Anfragen verwendet wurde. Daraus lässt sich die allgemeine *Wahrscheinlichkeit* des Ortes ableiten, um für zukünftige Anfragen Orte mit ähnlicher Wahrscheinlichkeit des realen Ortes auswählen zu können. Jeder Ort kann mit einem beliebigen Häufigkeits-Wert initialisiert werden. Nach jeder Ausführung von Lo*k*A wird der Häufigkeits-Wert des realen Ortes und der $k-1$ gewählten Dummy-Orte um 1 erhöht.

Die Liste wird fortlaufend nach den gegebenen Häufigkeits-Werten der Orte (aufsteigend) sortiert. Objekte mit gleichem Wert werden zufällig angeordnet. Bei einer neuen Anfrage mit echtem Ort r sollen aus den n gegebenen Orten $k-1$ Dummy-Orte gewählt werden. Dazu wird r aus L entfernt, damit eine Liste mit $n-1$ möglichen Dummy-Kandidaten entsteht.

Um diese Kandidaten auszuwählen, wird das Intervall $[0;1]$ gleichmäßig in $n-1$ Teilintervalle aufgeteilt. Das i-te Intervall $I(i)$ ist gegeben durch

$$I(i) = [\frac{i-1}{n-1}; \frac{i}{n-1}], 1 \leq i \leq n-1$$

Die $n-1$ Kandidaten-Elemente werden den $n-1$ Intervallen in ihrer jeweiligen Reihenfolge zugeordnet (das erste Element entspricht dem ersten Intervall, das zweite Element dem Zweiten, ...). Mit I_d wird das Intervall bezeichnet, welches dem Ort d zugeordnet ist.

Die Dummy-Orte werden nach einer auf dem Intervall $[0;1]$ abgeschlossenen Wahrscheinlichkeitsverteilung ausgewählt. Dafür wurde die *Beta-Verteilung* gewählt. Eine Beta-Verteilung ist eine stetige Wahrscheinlichkeitsverteilung mit zwei Parametern α und β. Sie ist mit der Dichtefunktion

$$f_{\alpha,\beta}(x) = \frac{x^{\alpha-1}(1-x)^{\beta-1}}{B(\alpha,\beta)}$$

mit

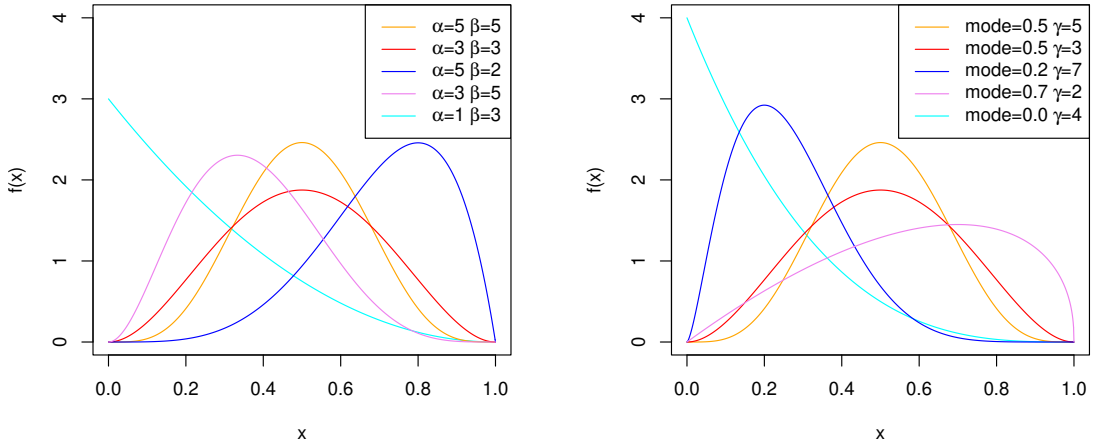
$$B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$$

definiert, wobei Γ die *Gammafunktion* ist. Durch die Parameter $\alpha, \beta > 0$ lässt sich die Form der Verteilungsfunktion bestimmen. Für diese Arbeit ist insbesondere der *Modus* der Beta-Verteilung relevant, der sich für $\alpha, \beta > 1$ an der Stelle $\frac{\alpha-1}{\alpha+\beta-2}$ befindet. Für $\alpha = 1, \beta > 1$ befindet er sich bei 0, für $\alpha > 1, \beta = 1$ bei 1.

Orte, die nahe r in der ursprünglichen Liste lagen, sollen mit einer höheren Wahrscheinlichkeit gewählt werden als Orte, die sich weiter entfernt befanden. Dazu wird die Beta-Verteilung so parametrisiert, dass sich ihr Modus an der Intervallgrenze der beiden Intervalle befindet, die den Orten zugeordnet sind, die in L direkt neben r lagen. Diese beiden Orte besitzen eine relativ hohe Wahrscheinlichkeit als Dummy-Orte ausgewählt zu werden. Im Ausnahmefall, dass r der erste Ort in L war, soll sich der Modus der Beta-Verteilung an der Stelle 0 befinden. Analog dazu soll er 1 sein, wenn r das letzte Element in L war.

Um die Beta-Verteilung aufzubauen, definieren wir die Funktion g wie folgt:

$$g(mode,\beta) = \frac{\beta \cdot mode - 2 \cdot mode + 1}{-mode + 1}$$



(a) Direkt mit α und β parametrisiert

(b) Durch geg. *mode* und γ Werte erzeugt

Abbildung 5.1: Dichtefunktionen unterschiedlich parametrisierter Beta-Verteilungen

g bestimmt für einen gegebenen β Wert und Modus ($mode \leq \frac{1}{2}$) den Wert α so, dass sich aus den Parametern eine entsprechend Beta-Verteilung erzeugen lässt.

Die Parameter α und β sind durch

$$\alpha = \begin{cases} g(mode, \gamma) & \text{falls } mode \leq \frac{1}{2} \\ \gamma & \text{sonst} \end{cases} \qquad \beta = \begin{cases} \gamma & \text{falls } mode \leq \frac{1}{2} \\ g(1 - mode, \gamma) & \text{sonst} \end{cases}$$

gegeben. Dabei ist $\gamma > 1$ ein Parameter von Lo*k*A, durch den sich die Form der Beta-Verteilung bestimmen lässt. Durch einen höheren γ-Wert ergibt sich eine spitzere Wahrscheinlichkeitsverteilung. Dadurch erhöht sich die Wahrscheinlichkeit, dass Orte gewählt werden, die nahe r in L lagen, gegenüber Orten, die sich weiter entfernt befanden.

Für Modi kleiner gleich $\frac{1}{2}$ definieren wir $\beta = \gamma$ und $\alpha = g(mode, \gamma)$. Bei Modi größer $\frac{1}{2}$ werden die Werte festgelegt auf $\beta = g(1 - mode, \gamma)$ und $\alpha = \gamma$. Dies entspricht der Erzeugung einer Beta-Verteilung mit Modus $1 - mode$ bei festem β Wert und anschließender Spiegelung an der Stelle $x = \frac{1}{2}$ durch Vertauschung von α und β. Die Dichtefunktion der Beta-Verteilung, die durch den obigen Prozess mit Parameter γ zu Ort r erzeugt wurde, wird als $beta_{r,\gamma}$ bezeichnet.

Die Wahrscheinlichkeit, dass ein bestimmter Ort d als Dummy-Ort gewählt wird, wird durch die von $beta_{r,\gamma}$ festgelegte Wahrscheinlichkeit des entsprechenden Intervalls I_d bestimmt:

$$P_{beta_{r,\gamma}}(d) = P_{beta_{r,\gamma}}(X \in I_d) = P_{beta_{r,\gamma}}(\frac{i-1}{n-1} \leq X \leq \frac{i}{n-1}) = \int_{\frac{i-1}{n-1}}^{\frac{i}{n-1}} beta_{r,\gamma}(x)dx$$

Mit den gegebenen Wahrscheinlichkeiten werden $k - 1$ Dummy-Orte zufällig generiert. Vereint mit r ergibt sich die Menge D, mit $|D| = k$.

Die Wahrscheinlichkeit, dass D unter Verwendung von $beta_{r,\gamma}$ in bestimmter Reihenfolge erzeugt wird, ist das Produkt $P_{beta_{r,\gamma}}(d_1) \cdot ... \cdot P_{beta_{r,\gamma}}(d_{k-1})$ der Wahrscheinlichkeiten der Dummy-Orte $d_1, ..., d_{k-1}$. Damit die Anonymität effektiv geschützt wird, sollte es ähnlich wahrscheinlich sein, D zu erzeugen, wenn ein anderes Element der Menge der echte Ort wäre.

Dazu wird wie folgt die Entropie von D bestimmt: Für jedes Element $a \in D$ wird angenommen, dass a der echte Ort ist. Entsprechend wird, wie beschrieben, a aus L entfernt, die übrigen Orte den $n-1$ Intervallen zugeordnet und die Verteilung $beta_{a,\gamma}$ erzeugt. Aus dieser lässt sich die Wahrscheinlichkeit bestimmen, dass D erzeugt würde, wenn a der reale Ort wäre. Diese ist (für eine beliebige, feste Reihenfolge) gegeben durch das Produkt der Wahrscheinlichkeiten der anderen Orte aus D im Bezug auf $beta_{a,\gamma}$:

$$P(D \mid r' = a) = \prod_{\substack{d \in D \\ d \neq a}} P_{beta_{a,\gamma}}(d)$$

Algorithmus 5.1 Lo*k*A Algorithmus

Input

L	List of locations, sorted by usage amount
r	The real location, $r \in L$
m	Number of generated sets
k	Parameter k
γ	The shape parameter

Output

RESULT	Resulting set, including r and $k-1$ dummies

```
maxEntropy ← −∞
RESULT ← NULL
for i = 0, ..., m − 1 do
    D ← generate(L, r, k, γ)        // Generate k − 1 random dummy elements with beta_{r,γ}
    D.add(r)
    PROBS ← ∅
    for a in D do
        p ← getProbability(D, a)        // Compute P(D | r' = a)
        PROBS.add(p)
    end for
    e ← entropy(PROBS)
    if e > maxEntropy then
        maxEntropy ← e
        RESULT ← D
    end if
end for
for l in RESULT do
    incrementFrequency(l)        // Increment the frequency value of the chosen locations by 1
end for
return RESULT
```

Wurden die bedingten Wahrscheinlichkeiten für alle Orte in D berechnet, lässt sich aus diesen die Entropie H_D der Menge bestimmen. Dazu werden die Wahrscheinlichkeiten $P(D \mid r' = a), \forall a \in D$ in die Entropiefunktion H eingesetzt:

$$H = -\sum_{a \in D} P(D \mid r' = a) \cdot \log_2 P(D \mid r' = a)$$

Diese Prozedur wird m mal wiederholt, wobei $m > 0$ ein weiterer Parameter von LokA ist. Ein höheres m erhöht die durchschnittliche Entropie der finalen Resultate auf Kosten eines höheren Berechnungsaufwandes. Sollte während des Prozesses eine Menge erzeugt werden, die Duplikate enthält, wird diese verworfen und erneut generiert. Für alle erzeugten Mengen $D_1, ..., D_m$ werden die Entropien $H_{D_1}, ..., H_{D_m}$ berechnet. Die Menge, welche die größte Entropie besitzt, wird schlussendlich als Resultat gewählt.

5.2 Evaluation

Für die Evaluation wird eine Liste L mit $n = 100$ Orten und willkürlich zugeordneten Häufigkeits-Werten definiert und nach diesen sortiert. Das Element bei Index 0 besitzt den Niedrigsten, das bei Index 99 den höchsten Häufigkeits-Wert.

Eine lokationsbasierte Anfrage soll mit Parameter $k = 3$ anonymisiert werden. Der reale Ort r befindet sich in der Liste bei Index 25. Es sollen $m = 20$ Mengen gebildet werden, von denen die Menge mit der größten Entropie als Resultat gewählt wird.

i	$D_i = \{r, d_1, d_2\}$	$P(D_i \mid r' = r)$	$P(D_i \mid r' = d_1)$	$P(D_i \mid r' = d_2)$	H_{D_i}
0	{25,34,13}	0,052%	0,013%	0,032%	0.011
1	{25,22,8}	0,026%	0,044%	0,044%	0.013
2	{25,41,44}	0,024%	0,045%	0,031%	0.012
3	**{25,33,27}**	**0,096%**	**0,081%**	**0,102%**	**0.028**
4	{25,28,46}	0,037%	0,048%	0,007%	0.01
5	{25,9,27}	0,033%	0,033%	0,023%	0.011
6	{25,33,35}	0,075%	0,089%	0,079%	0.025
7	{25,26,15}	0,079%	0,072%	0,076%	0.024
8	{25,18,54}	0,012%	0,006%	<0,001%	0.002
9	{25,38,44}	0,03%	0,058%	0,028%	0.013
10	{25,35,32}	0,078%	0,078%	0,093%	0.025
11	{25,34,22}	0,091%	0,053%	0,081%	0.023
12	{25,23,47}	0,034%	0,028%	0,002%	0.008
13	{25,40,12}	0,032%	0,001%	0,014%	0.006
14	{25,42,10}	0,02%	<0,001%	0,008%	0.004

Tabelle 5.1: Generierte Mengen mit Parameter $\gamma = 10$

i	$D_i = \{r, d_1, d_2\}$	$P(D_i \mid r' = r)$	$P(D_i \mid r' = d_1)$	$P(D_i \mid r' = d_2)$	H_{D_i}
0	{25,10,46}	0,021%	0,02%	0,005%	0.006
1	{25,4,23}	0,017%	0,033%	0,02%	0.008
2	{25,57,62}	0,009%	0,015%	0,012%	0.005
3	{25,43,32}	0,027%	0,024%	0,029%	0.01
4	{25,66,34}	0,013%	0,006%	0,016%	0.004
5	{25,33,5}	0,019%	0,01%	0,026%	0.007
6	{25,44,48}	0,021%	0,025%	0,022%	0.008
7	**{25,13,31}**	**0,029%**	**0,031%**	**0,023%**	**0.01**
8	{25,17,77}	0,006%	0,005%	0,001%	0.002
9	{25,52,63}	0,01%	0,017%	0,011%	0.005
10	{25,41,48}	0,022%	0,026%	0,021%	0.008
11	{25,26,46}	0,026%	0,027%	0,017%	0.008
12	{25,68,26}	0,012%	0,003%	0,012%	0.004
13	{25,56,8}	0,014%	0,001%	0,013%	0.004
14	{25,6,59}	0,011%	0,011%	0,001%	0.003

Tabelle 5.2: Generierte Mengen mit Parameter $\gamma = 3$

i	$D_i = \{r, d_1, d_2\}$	$P(D_i \mid r' = r)$	$P(D_i \mid r' = d_1)$	$P(D_i \mid r' = d_2)$	H_{D_i}
0	{25,18,29}	0,198%	0,053%	0,046%	0.029
1	{25,22,15}	0,07%	0,213%	0,068%	0.033
2	{25,31,32}	0,178%	0,315%	0,251%	0.064
3	{25,24,28}	0,513%	0,471%	0,398%	0.107
4	{25,33,26}	0,234%	0,089%	0,288%	0.054
5	{25,24,15}	0,078%	0,118%	0,041%	0.024
6	{25,28,29}	0,42%	0,495%	0,442%	0.106
7	{25,19,24}	0,314%	0,264%	0,388%	0.08
8	{25,37,21}	0,061%	<0,001%	0,014%	0.008
9	{25,33,30}	0,169%	0,173%	0,344%	0.06
10	**{25,27,29}**	**0,45%**	**0,519%**	**0,426%**	**0.108**
11	{25,23,29}	0,449%	0,344%	0,253%	0.085
12	{25,34,23}	0,178%	0,017%	0,103%	0.029
13	{25,17,31}	0,109%	0,015%	0,007%	0.013
14	{25,16,32}	0,062%	0,005%	0,001%	0.008

Tabelle 5.3: Generierte Mengen mit Parameter $\gamma = 50$

Tabelle 5.1 zeigt Mengen, die von Lo*k*A mit Parameter $\gamma = 10$ produziert wurden. Die Prozentwerte der Wahrscheinlichkeiten sowie die Entropie-Werte wurden auf drei Nachkommastellen gerundet. Es wurde die Menge D_4 ausgewählt, da sie von den generierten Mengen die größte Entropie besitzt. Die bedingten Wahrscheinlichkeiten $P(D_4 \mid r' = r) = 0,096\%$, $P(D_4 \mid r' = d_1) = 0,081\%$ und $P(D_4 \mid r' = d_2) = 0,102\%$ liegen nah beieinander, daher ist es schwer, in dieser Menge den realen Ort zu identifizieren.

Ein Beispiel für eine Menge mit geringer Entropie ist die Menge D_8 aus selbiger Tabelle. Bei dieser ergeben sich die bedingten Wahrscheinlichkeiten $P(D_8 \mid r' = r) = 0,012\%$, $P(D_8 \mid r' = d_1) = 0,006\%$ und $P(D_8 \mid r' = d_2) < 0,001\%$. Ohne zu wissen, dass r der reale Ort ist, kann d_2 bereits mit einer relativ hohen Sicherheit ausgeschlossen werden. Die in dieser Menge vorhandenen Dummy-Elemente eignen sich daher schlecht, um r zu anonymisieren.

In Tabelle 5.2 sind Mengen dargestellt, die mit Parameter $\gamma = 3$ erzeugt wurden. Es lässt sich beobachten, dass in diesem Fall vermehrt Dummy-Orte gewählt werden, die (im Bezug auf die Position der Elemente in L) weiter entfernt von r liegen. Dies erhöht die Unvorhersehbarkeit des Algorithmus, da die Wahrscheinlichkeit, dass weiter entfernte Orte als Dummy-Elemente ausgewählt werden, steigt. Dies geschieht allerdings auf Kosten der durchschnittlichen Entropie der produzierten Mengen. In diesem Beispiel besitzt die gewählte Menge eine im Vergleich zum vorherigen Beispiel geringere Entropie von 0.01.

Das Gegenteil bewirkt ein höherer γ Wert. Mengen, die mit Parameter $\gamma = 50$ generiert wurden, sind in Tabelle 5.3 präsentiert. Ersichtlich ist, dass in diesem Fall häufig nähere Elemente als Dummy-Elemente gewählt wurden. Dadurch steigen wiederum die durchschnittlichen Entropien der gebildeten Mengen, in diesem Fall besitzt die Menge mit der höchsten Entropie einen Wert von 0.108.

6 CaDaA - Anonymisierung von Kameradaten

Wie bereits von Xiong et al. erwähnt, können die von einem CC beziehungsweise SDC produzierten Kamerabilder sensible Hintergrundinformationen enthalten, über die Rückschlüsse auf den aktuellen Ort des Autos geführt werden können [XLHC19]. Es existiert bereits Software, mit denen sich der Aufnahmeort von Bildern durch die in diesen enthaltenen Informationen bestimmen lässt. Das von Google entwickelte PlaNet macht dies möglich, indem ein neuronales Netzwerk mit mehreren Millionen von Bildern trainiert wurde [WKP16]. Damit lässt sich durch Informationen wie die sich im Hintergrund der Bilder befindlichen Gebäude der Aufnahmeort des Bildes bestimmen. Um dem entgegenzuwirken, wird *Camera Data Anonymization* (CaDaA) vorgestellt, ein Konzept, um entsprechende Kamerabilder auf die nötigsten Informationen zu reduzieren.

6.1 Konzept

Aus einem Original-Bild I soll ein (gleichgroßes) Bild R entstehen, in dem ungewollte Informationen bestimmt und unkenntlich gemacht werden. Dazu wird das Input-Bild zunächst durch Bildsegmentierung in Klassen aufgeteilt. Dabei wird jedem Pixel eine eindeutige Klasse zugeordnet (z.B. *Auto*, *Straße*, etc.). Mit $class(i,j)$ bezeichnen wir die Klasse, die dem Pixel an der Stelle (i,j) im Input-Bild I zugeordnet wurde. Zu jeder Klasse k wird ein Zahlenwert als Priorität festgelegt. Ein höherer Wert bedeutet eine höhere Wichtigkeit der Klasse. Klassen, die als weniger wichtig erachtet werden, werden einem niedrigerem Zahlenwert zugeordnet. Sei $cp(k)$ der Prioritätswert, welcher der Klasse k zugeordnet ist. Des Weiteren definieren wir

$$p(i,j) = (cp \circ class)(i,j)$$

als die *Priorität* des Pixels (i,j).

Das Input-Bild I wird zunächst komplett mit einer beliebig definierten Bildbearbeitungsfunktion f verarbeitet, die ein gleichgroßes Bild Q erzeugt (z.B. ein Verpixelungs-Algorithmus). $I(i,j)$ bezeichne den *Farbwert* des Pixels an der Stelle (i,j) im originalen Bild I. $Q(i,j)$ und $R(i,j)$ seien analog definiert. Damit nur ungewollte Hintergrundinformationen aus dem finalen Resultat entfernt werden, aber wichtige Informationen erhalten bleiben, muss für jeden Pixel des endgültigen Bildes R entschieden werden, ob dessen Farbwert den Wert des entsprechenden Pixels aus I oder aus Q erhalten soll.

Um diese Entscheidung zu treffen, wird ein Parameter *treshold* eingeführt. Alle Pixel, deren Prioritätswert unter dem Schwellwert liegt, werden mit dem Farbwert des entsprechenden Pixels des verarbeiteten Bildes Q belegt. Alle weiteren Pixel werden auf den entsprechenden Farbwert aus I gesetzt. Dadurch bleiben dem Bild Informationen enthalten, die als wichtig erachtet werden, während weniger wichtige Informationen unkenntlich gemacht werden. Je nach Situation kann der

Schwellwert erhöht beziehungsweise gesenkt werden, um mehr beziehungsweise weniger Klassen aus dem resultierenden Bild R zu verschleiern. Formal ausgedrückt wird der Wert $R(i,j)$ bestimmt durch

$$R(i,j) = \begin{cases} Q(i,j) & \text{falls } p(i,j) < treshold \\ I(i,j) & \text{sonst} \end{cases}$$

Durch Auswertung von p für jeden Pixel des Originalen-Bildes I ergibt sich das finale Bild R.

Algorithmus 6.1 CaDaA Algorithmus

Input

I	Image of size $n \times m$
cp	The class priorities
treshold	The threshold value
f	Image processing function

Output

R	Processed image of size $n \times m$

$class \leftarrow segmentate(I)$ // Compute and store the class of each pixel with image segmantation
$Q \leftarrow f(I)$
$R \leftarrow \emptyset$
for $i = 0, ..., n-1$ **do**
  for $j = 0, ..., m-1$ **do**
    $pixelClass \leftarrow class(i,j)$
    $pixelPrio \leftarrow cp(pixelClass)$ // Compute priority for each pixel
    if $pixelPrio < treshold$ **then**
      $R(i,j) \leftarrow Q(i,j)$
    else
      $R(i,j) \leftarrow I(i,j)$
    end if
  end for
end for
return R

6.2 Evaluation

Für die Implementierung wird ein Bildsegmentierungs-Algorithmus verwendet, der auf einem trainierten neuronalem Netzwerk aufbaut, um Bilder in insgesamt 20 verschiedene Klassen aufzuteilen [Add21].

Zunächst müssen die Prioritäten der Klassen sinnvoll festgelegt werden. Klassen, die andere Verkehrsteilnehmer, wie zum Beispiel Autos und Busse beschreiben, wurden sehr hohe Prioritäten zugeordnet. Diese sollten nur gefiltert werden, wenn ein sehr hoher Schwellwert gewählt wird. Allgemein wurden Klassen, die für die Dienste eines CCs von großer Bedeutung sind, mit hohen Priotitätswerten belegt. Zum Beispiel ist es für autonomes Fahren wichtig, dass Informationen wie andere Verkehrsteilnehmer, aber auch die Straße, Personen etc. erkennbar sind.

Hintergrundinformationen wie Gebäude, Vegetation und der Himmel wurden hingegen mit niedrigen Prioritäten versehen. Diese sollen in der Regel aus den Bildern entfernt werden, da sie Rückschlüsse auf den Ort eines CCs zulassen können. Die Klassen und die für die Evaluation gewählten Prioritäten sind in Tabelle 6.1 dargestellt.

Klasse	Priorität
Unbestimmt	5
Straße	8
Bürgersteig	8
Gebäude	4
Wand	3
Zaun	2
Stange	2
Ampel	6
Verkehrsschild	6
Vegetation	1
Terrain	1
Himmel	0
Person	7
Fahrer	9
Auto	9
LKW	9
Bus	9
Zug	9
Motorrad	9
Fahrrad	7

Tabelle 6.1: Klassen-Prioritäten

CaDaA wurde auf dem Argoverse 1 Datensatz ausgewertet [CLS+19]. In Abbildung 6.1 ist zu sehen, wie CaDaA auf ein beispielhaftes Bild, das von der vorderen Kamera eines Autos aufgenommen wurde, angewandt wurde. Als *treshold* Parameter wird in diesem Beispiel der Wert 6 gewählt. Die Verarbeitungsfunktion f ist in diesem Beispiel eine simple Verpixelungs-Funktion.

Es ist zu beobachten, dass nur die Informationen unkenntlich gemacht wurden, deren Priorität unter dem gegebenen Schwellwert liegt. In diesem Beispiel sind das die Klassen *Unbestimmt*, *Gebäude*, *Wand*, *Zaun*, *Stange*, *Vegetation*, *Terrain* und *Himmel*. Alle Pixel, die in eine der genannten Klassen geordnet wurden, stammen aus dem verarbeiteten Bild Q. Die Pixel, denen Klassen zugeordnet wurden, deren Priorität nicht unter dem Schwellwert liegt, blieben im Endresultat unverändert. In diesem Beispiel sind das die Klassen *Straße*, *Bürgersteig*, *Ampel*, *Verkehrsschild*, *Person*, *Fahrer*, *Auto*, *LKW*, *Bus*, *Zug*, *Motorrad* und *Fahrrad*. Durch Änderung des Schwellwertes lässt sich situationsabhängig bestimmen, welche Klassen in den Bildern unkenntlich gemacht werden sollen.

Die Auswirkungen unterschiedlich gewählter Schwellwerte und Verarbeitungsfunktionen auf die von CaDaA produzierten Ergebnisse werden in Abbildung 6.2 dargestellt. Die Bilder **(a)** bis **(d)** zeigen, wie sich die Ergebnisse mit unterschiedlich gewählten Verarbeitungsfunktionen bei festem Schwellwert unterscheiden. In den Bildern **(e)** bis **(h)** werden Änderungen des Schwellwertes bei einheitlicher Verarbeitungsfunktion dargestellt.

(a) Originales Bild **(b)** Mit CaDaA verarbeitetes Bild

Abbildung 6.1: Beispiel eines von CaDaA verarbeiteten Bildes

(a) Original

(b) $treshold = 6; f = pixelate()$

(c) $treshold = 6; f = blur()$

(d) $treshold = 6; f = blacken()$

(e) Original

(f) $treshold = 3; f = pixelate()$

(g) $treshold = 5; f = pixelate()$

(h) $treshold = 9; f = pixelate()$

Abbildung 6.2: Resultate von CaDaA bei unterschiedlich gewählten Parametern

7 SpAn - Anonymisierung von Tempodaten

Dieses Kapitel beschäftigt sich mit der Anonymisierung von Geschwindigkeitsdaten. Dies ist zum Beispiel sinnvoll, wenn Benutzer nicht möchten, dass Dienstanbieter wie Versicherungsfirmen Informationen über deren Fahrverhalten erhalten sollen. Im beispielhaften Fall einer überhöhten Geschwindigkeit sollen möglichst realistische Geschwindigkeitswerte in einem festgelegten Tempobereich zufällig generiert werden. Mit *Speed Anonymization* (SpAn) wird ein Konzept entworfen, welches einen realitätsnahen Geschwindigkeitsverlauf in einem bestimmten Bereich simuliert.

7.1 Konzept

Aus einem realem Geschwindigkeitswert *speed* soll zum aktuellen Zeitpunkt t_0 ein simulierter Geschwindigkeitswert erzeugt werden. Dieser Wert soll in jedem Fall zwischen den beiden festgelegten Grenzen *min* und *max* liegen, mit $min < max$. Es werden allgemein keine Werte erzeugt, die außerhalb dieser beiden Grenzen liegen.

Mit $lastValue \in [min; max]$ wird der Wert bezeichnet, der von SpAn bei der letzten Ausführung zum Zeitpunkt t_{-1} ausgegeben wurde. Alle generierten Werte sind abhängig von den jeweilig zuletzt generierten Werten und deren Zeitpunkten. Zum Zeitpunkt der ersten Ausführung gilt $lastValue = fit(speed, min, max)$ und $t_{-1} = t_0$. Dabei ist die Funktion fit definiert als

$$fit(value, min, max) = \begin{cases} max & \text{falls } value > max \\ value & \text{falls } value \leq max \text{ und } value \geq min \\ min & \text{sonst} \end{cases}$$

Zusätzlich wird eine Zeitdauer $relax > 0$ und ein Abweichungsfaktor $deviation > 0$ festgelegt. Durch diese beiden Parameter wird abhängig vom Zeitraum zwischen dem Letzten und dem aktuellen Zeitpunkt bestimmt, wie viel der aktuell zu generierende Wert maximal vom Letzten abweichen darf. Dadurch soll eine realistische Wertegenerierung gesichert werden. Es wäre zum Beispiel sehr unrealistisch, wenn innerhalb von $100ms$ ein Geschwindigkeitsunterschied von $\pm 10km/h$ auftreten würde. Dazu wird eine *maximale Abweichung* bestimmt, gegeben durch

$$maxDeviation = \frac{t_0 - t_{-1}}{relax} \cdot deviation$$

Daraus ergibt sich die untere und obere Grenze der aktuellen Iteration:

$$lower = max(min, lastValue - maxDeviation)$$

$$upper = min(max, lastValue + maxDeviation)$$

Der erzeugte Wert weicht um maximal den Wert *maxDeviation* vom Vorherigen ab, ist aber niemals größer als die allgemeine Grenze *max* beziehungsweise kleiner als *min*. Im Fall $t_{-1} = t_0$ ergibt sich $lower = upper = lastValue$, dabei wird der entsprechende Wert sofort als Resultat gesetzt.

Für die Wahrscheinlichkeitsverteilung bestimmen wir einen Pivot-Wert $pivot \in [lower, upper]$, gegeben durch

$$pivot = fit(speed, lower, upper)$$

Werte nahe des Pivot-Wertes sollen mit einer höheren Wahrscheinlichkeit erzeugt werden als weiter entfernte Werte. Dazu wird, wie bereits in Kapitel 5 beschrieben, eine Beta-Verteilung mit festgelegtem Modus und Parameter $\gamma > 1$ erzeugt. Der Modus wird bestimmt durch

$$mode = \frac{pivot - lower}{upper - lower}$$

Algorithmus 7.1 SpAn Algorithmus

Input

speed	Current (actual) speed
t_0	Current timestamp (in ms)
lastValue	Last generated value
t_{-1}	Last timestamp (in ms)
min	General minimum value
max	General maximum value
relax	Reference recovery duration (in ms)
deviation	deviation factor
γ	The shape parameter

Output

result	Generated speed value

$sinceLast \leftarrow t_0 - t_{-1}$
$factor \leftarrow \frac{sinceLast}{relax}$
$maxDeviation \leftarrow factor \cdot deviation$
$lower \leftarrow max(min, lastValue - maxDeviation)$
$upper \leftarrow min(max, lastValue + maxDeviation)$
$result \leftarrow NULL$
if $lower \neq upper$ **then**
    $pivot \leftarrow fit(speed, lower, upper)$
    $mode \leftarrow \frac{pivot - lower}{upper - lower}$
    $x \leftarrow generate(pivot, mode, \gamma)$ // Generate a random value $x \in [0; 1]$ with $beta_{\gamma}^{mode}$
    $result \leftarrow lower + (upper - lower) \cdot x$
else
    $result \leftarrow lastValue$ // When $t_0 = t_{-1}$
end if
$lastValue \leftarrow result$
$t_{-1} \leftarrow t_0$
return *result*

Er entspricht der relativen Position von *pivot* im Intervall $[lower; upper]$. Mit $beta_{\gamma}^{mode}$ wird die durch gegebenen Modus und γ Wert erzeugte Beta-Verteilung beschrieben.

Anhand der durch $beta_{\gamma}^{mode}$ gegebenen Wahrscheinlichkeitsverteilung wird ein zufälliger Wert $x \in [0; 1]$ (mittels Inversionsmethode) erzeugt. Das Resultat ist dann

$$result = lower + (upper - lower) \cdot x$$

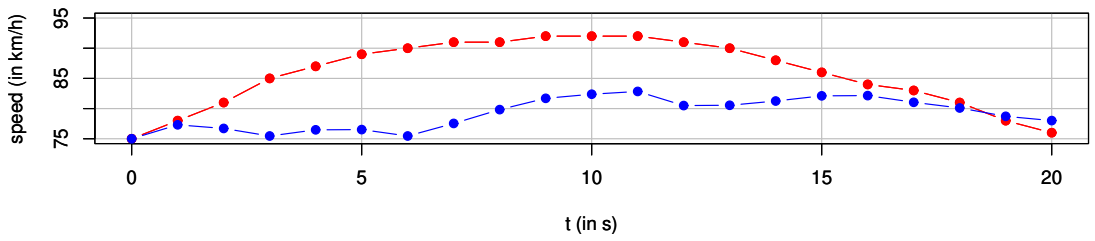
Dabei gilt $result \in [lower; upper]$. Dies ist der Wert, der schlussendlich zurückgegeben wird.

7.2 Evaluation

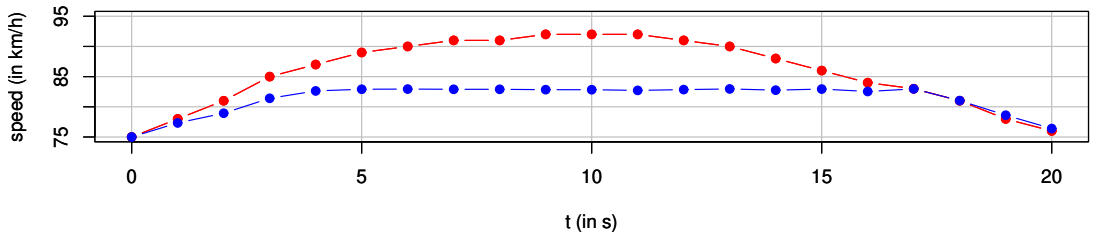
Es sollen Geschwindigkeitsdaten ab einem bestimmten Zeitpunkt $t = 0s$ anonymisiert werden. In diesem Beispiel gehen wir davon aus, dass die Geschwindigkeitsdaten sekündlich verarbeitet werden. Als Parameter werden $min = 75$, $max = 83$, $relax = 1000ms$ und $deviation = 2.5$ gewählt. Jeder erzeugte Wert weicht daher maximal um

$$maxDeviation = \frac{1000ms}{1000ms} \cdot 2.5km/h = 2.5km/h$$

vom Vorherigen ab.



(a) $\gamma = 1.5$



(b) $\gamma = 10$

Abbildung 7.1: Resultate von SpAn bei unterschiedlich gewähltem γ Parameter

In Abbildung 7.1 sind zwei durch SpAn erzeugte Geschwindigkeitsverläufe mit unterschiedlichem γ Parameter grafisch dargestellt. Die roten Punkte stellen den echten Verlauf dar. Durch die blauen Punkte werden die generierten Verläufe abgebildet.

Die erste Anwendung von SpAn findet jeweils zum Zeitpunkt $t = 0$ statt. Dabei gilt $result = fit(speed, min, max) = speed$, da sich der aktuelle Geschwindgkeitswert noch unter dem Maximum befindet. Alle weiteren Werte werden maximal $2.5km/h$ voneinander entfernt erzeugt, wobei sich jeweils in Richtung des Pivot-Wertes orientiert wird:

$$pivot = fit(speed, lower, upper)$$

mit

$$lower = max(min, lastValue - 2.5km/h)$$

$$upper = min(max, lastValue + 2.5km/h)$$

Es ist zu beobachten, dass sich durch einen höheren γ Wert stärker an diesem Pivot-Wert orientiert wird als bei einem niedrigerem γ. Durch einen niedrigeren Wert werden die Ergebnisse unvorhersehbarer und willkürlicher, während ein höherer Wert eine stabilere Linie erzeugt.

8 Zusammenfassung und Ausblick

Diese Bachelorarbeit setzte sich mit der Problematik des situationsabhängigen Schutzes von Daten, die von einem CC erzeugt und verarbeitet werden, auseinander. Dabei wurde mit PLEvS ein Konzept vorgestellt, mit dem sich benutzerdefiniert Situationen erstellen lassen können, die von definierten Konditionen und Attributen abhängig sind. Dazu wurde das Privacy-Level eingeführt, welches durch den aktuellen Zustand der definierten Situationen für einen bestimmten Anfragetyp die anzuwendende Anonymisierungsfunktion bestimmt.

Des Weiteren wurde mit Lo*k*A ein Algorithmus entworfen, der Ortsdaten effektiv anonymisiert. Durch Auswahl von Dummy-Orten abhängig der allgemeinen Wahrscheinlichkeit des realen Ortes und Maximierung der Entropie von bedingten Wahrscheinlichkeiten wurde sichergestellt, dass der echte Ort unter den Dummy-Orten nicht identifiziert werden kann.

Durch CaDaA wurde ein Konzept eingeführt, um Daten aus Kamerabildern zu entfernen, aus denen sensible Informationen abgeleitet werden können. Der Algorithmus lässt sich situationsabhängig parametrisieren, um festzulegen, welche Daten entfernt und welche erhalten bleiben sollen.

Zum Schluss wurde SpAn vorgestellt, ein Konzept, um einen realistischen Geschwindigkeitsverlauf realistisch zu simulieren, um Informationen über das Fahrverhalten des Benutzers zu anonymisieren. Dabei wurde darauf geachtet, keine unrealistischen Geschwindigkeitssprünge zu erzeugen, wodurch die simulierten Geschwindigkeiten identifiziert werden können.

Die vorgestellten Konzepte sollen zum Thema des situationsabhängigen Schutzes der Privatsphäre von Personen in CC-Szenarien beitragen.

Ausblick

Der Schutz der Privatsphäre in CCs beziehungsweise SDCs ist ein Thema, dessen Wichtigkeit in den kommenden Jahren vermutlich kontinuierlich weiter steigen wird. Mit steigender Anzahl an Diensten, die von solchen Autos angeboten werden, wird es auch in Zukunft eine große Herausforderung sein, die Privatsphäre der Benutzer effektiv zu schützen.

Alle in dieser Bachelorarbeit gezeigten Konzepte bieten sich an, um erweitert und weiter verbessert zu werden. Durch Kombination mit anderen existierenden und zukünftigen Konzeptideen, kann die Privatsphäre von Benutzern zukünftig noch effektiver und effizienter geschützt werden.

Literaturverzeichnis

[Add21] S.-C. Addison. *Automatic Addison*. Feb. 2021. URL: https://automaticaddison.com/how-to-detect-objects-using-semantic-segmentation/ (zitiert auf S. 42).

[AKA20] A. Akca, I. Kara, M. Aydos. „Privacy, Security and Legal Aspects of Autonomous Vehicles“. In: *International Conference on Materials Science, Mechanical and Automotive Engineerings and Technology (IMSMATEC'20)*. Juli 2020 (zitiert auf S. 20).

[CLS+19] M.-F. Chang, J. W. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan, J. Hays. „Argoverse: 3D Tracking and Forecasting with Rich Maps“. In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019 (zitiert auf S. 43).

[DSR19] R. Dave, E. Sowells-Boone, K. Roy. „Efficient Data Privacy and Security in Autonomous Cars“. In: *Journal of Computer Sciences and Applications* 7 (Mai 2019), S. 31–36. DOI: 10.12691/jcsa-7-1-5 (zitiert auf S. 19).

[Eur16] European Commission. *REGULATION (EU) 2016/679 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL*. 2016. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679 (zitiert auf S. 20).

[HHM+17] M. Hüffmeyer., P. Hirmer., B. Mitschang., U. Schreier., M. Wieland. „SitAC – A System for Situation-aware Access Control - Controlling Access to Sensor Data“. In: *Proceedings of the 3rd International Conference on Information Systems Security and Privacy - ICISSP,* INSTICC. SciTePress, 2017, S. 113–125. ISBN: 978-989-758-209-7. DOI: 10.5220/0006186501130125 (zitiert auf S. 20).

[NLZ+14] B. Niu, Q. Li, X. Zhu, G. Cao, H. Li. „Achieving k-anonymity in privacy-aware location-based services“. In: *IEEE INFOCOM 2014 - IEEE Conference on Computer Communications*. 2014, S. 754–762. DOI: 10.1109/INFOCOM.2014.6848002 (zitiert auf S. 17, 18, 21, 33).

[PBCK19] L. Porzi, S. R. Bulò, A. Colovic, P. Kontschieder. „Seamless Scene Segmentation“. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, S. 8269–8278. DOI: 10.1109/CVPR.2019.00847 (zitiert auf S. 22).

[WCY20] J. Wang, Z. Cai, J. Yu. „Achieving Personalized *k*-Anonymity-Based Content Privacy for Autonomous Vehicles in CPS“. In: *IEEE Transactions on Industrial Informatics* 16.6 (2020), S. 4242–4251. DOI: 10.1109/TII.2019.2950057 (zitiert auf S. 19).

[WKP16] T. Weyand, I. Kostrikov, J. Philbin. „PlaNet - Photo Geolocation with Convolutional Neural Networks“. In: *Computer Vision – ECCV 2016*. Springer International Publishing, 2016, S. 37–55. DOI: 10.1007/978-3-319-46484-8_3. URL: https://doi.org/10.1007%2F978-3-319-46484-8_3 (zitiert auf S. 41).

[XLHC19] Z. Xiong, W. Li, Q. Han, Z. Cai. „Privacy-Preserving Auto-Driving: A GAN-Based Approach to Protect Vehicular Camera Data“. In: *2019 IEEE International Conference on Data Mining (ICDM)*. 2019, S. 668–677. DOI: 10.1109/ICDM.2019.00077 (zitiert auf S. 22, 41).

[Yan20] B. Yankson. „Autonomous Vehicle Security Through Privacy Integrated Context Ontology(PICO)“. In: *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. 2020, S. 4372–4378. DOI: 10.1109/SMC42975.2020.9283180 (zitiert auf S. 19).

[YZK+17] J. Yu, B. Zhang, Z. Kuang, D. Lin, J. Fan. „iPrivacy: Image Privacy Protection by Identifying Sensitive Objects via Deep Multi-Task Learning“. In: *IEEE Transactions on Information Forensics and Security* 12.5 (2017), S. 1005–1016. DOI: 10.1109/TIFS.2016.2636090 (zitiert auf S. 22).

Alle URLs wurden zuletzt am 04. 04. 2022 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

Ort, Datum, Unterschrift