Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Janek Tichy

Simulationsbasierte Analyse von C2X Kommunikation zur Vermeidung von Kollisionen unterschiedlicher Verkehrsteilnehmer

Fakultät Technik und Informatik
Studiendepartment Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Janek Tichy

Simulationsbasierte Analyse von C2X Kommunikation zur Vermeidung von Kollisionen unterschiedlicher Verkehrsteilnehmer

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung

im Studiengang Bachelor of Science Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Korf
Zweitgutachter: Prof. Dr. Becke

Eingereicht am: 14. November 2016

Janek Tichy

Thema der Arbeit

Simulationsbasierte Analyse von C2X Kommunikation zur Vermeidung von Kollisionen unterschiedlicher Verkehrsteilnehmer

Stichworte

C2C, C2X, OMNeT++, SUMO, Veins, Kollisionsvermeidung, Simulation

Kurzzusammenfassung

Die vorliegende Bachelorarbeit befasst sich mit der Leistungsfähigkeit eines Fahrerassistenzsystems, das einen Datenaustausch unter Verkehrsteilnehmern ermöglicht um Kollisionen zu vermeiden. Die Empfangenen Nachrichten werden analysiert und der Fahrzeugführer wird benachrichtigt, wenn eine bevorstehende Kollision erkannt wird. Die Leistungsfähigkeit des Assistenzsystems wird simulationsbasiert erforscht unter den Einflüssen von dem Sendeintervall der Nachricht, dem Mindestabstand zum vorausfahrenden Verkehrsteilnehmer, der Reaktionszeit des Fahrzeugführers, der Positionsungenauigkeit und Ausnahmefahrern, die das Assistenzsystem nicht nutzen und Kreuzungen unbeachtet überqueren.

Janek Tichy

Title of the paper

Simulation oriented investigation of C2X communication to avoid collision of road user

Keywords

C2C, C2X, OMNeT++, SUMO, Veins, Collision Avoidance, Simulation

Abstract

The bachelor thesis deals with the performance of advanced driver assistance system, which enables the road user to exchange information among each other in order to avoid collision. The received data is analyzed and the driver gets notified if collisions get recognized. The assistance system is simulation based investigated under the influence of the interval the data is send, the minimum distance to the car ahead, the reaction time of the driver, the position imprecision and exceptional driver, who do not use the assistance system and just cross intersections.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

Listings

1 Einleitung

Verkehrsmittel wie Flugzeuge und Kraftfahrzeuge haben das moderne Leben revolutioniert, indem lange Strecken komfortabel zurückgelegt werden und dabei die Sicherheit der Reisenden höchste Priorität hat. Flugzeuge gelten als die sichersten Verkehrsmittel, indem sie mit Assistenzsystemen ausgestattet sind, die nicht nur die Motoren überwachen, oder dafür sorgen, dass die Triebwerke effizient arbeiten, sondern auch mit Systemen, die eine Kommunikation unter den Flugzeugen herstellt, um im Notfall dem Piloten einen Höhenwechsel vorzuschlagen. Systeme die den Fahrzeugführer rechtzeitig vor einer Gefahr warnen, tragen dazu bei, dass der Straßenverkehr sicherer wird und weniger Sach- und Personenschäden entstehen. Trotz alledem sind vergleichbare Assistenzsysteme nicht für den Straßenverkehr verfügbar, obwohl in Deutschland im Jahr 2015 397.219 Menschen durch einen Verkehrsunfall zu Schaden gekommen- und darunter 3.475 Menschen gestorben sind, welches einen Zuwachs von 2,9% zum Vorjahr entspricht[1]. Laut Bundesanstalt für Straßenwesen verursachen Personen- und Sachschäden Volkswirtschaftliche Kosten von über 30 Milliarden Euro[2].

Das in dieser Bachelorarbeit vorgestellte Assistenzsystem wird von Fahrzeugführern auf dem privaten Mobiltelefon ausgeführt. Es stellt die aktuelle Position, Beschleunigung, Fahrzeuglänge, Geschwindigkeit, ID, Fahrrichtung und Zeit zur Verfügung und wird von jedem Assistenzsystem in Reichweite empfangen. Auf Basis der Nachrichtentupel der in Reichweite liegenden Assistenzsysteme wird der Fahrzeugführer auf Situationen hingewiesen die zu Kollisionen führen. Die zeitkritische Anwendung, die einen Datenaustausch zwischen Verkehrsteilnehmern herstellt, unterliegt physikalischen Anforderungen um ein zufriedenstellendes Ergebnis zu erzielen. Zu diesen Anforderungen gehört der Datenaustausch, der schnell genug sein muss um auf Statusänderungen eines Verkehrsteilnehmers angemessen zu reagieren. Daraus folgen die Probleme wie oft pro Sekunde ein Datenaustausch stattfinden muss und wie viele Sender in einem Netzwerk existieren können, ohne einen Stau auf dem Übertragungsmedium zu verursachen. Jeder Teilnehmer muss in einen definierten Zeitintervall ein Statusupdate versenden und blockiert dadurch eine Signallaufzeit das Medium. Die Anzahl der Sender im Netzwerk hat Einfluss auf

die Anwendbarkeit des Systems, denn durch die Zunahme von Sendern im Netzwerk steigt auch die Anwendbarkeit des Systems, sowie die Verlässlichkeit der Daten. Wie schnell kann eine Aktualisierung der Positionsdaten erfolgen? Wie genau sind Positionsangaben und wie kann auf eine Ungenauigkeit reagiert werden? Es ist auszuarbeiten welche Protokolle für den Anwendungsfall geeignet sind und, ob zusätzliche Features der jeweiligen Protokolle benötigt werden, wie z.B. Congession Control von TCP. Bei dem Datenaustausch findet ein Wettkampf um das Übertragungsmedium statt. Der Mac Algorithmus der den Zugriff auf das Medium definiert stellt sicher, dass der Zugriff fair ist und alle Teilnehmer gleichmäßig darauf Zugriff haben. Weiterhin muss der Algorithmus dezentral ablaufen, da es keine zentrale Sendeeinheit existiert.

Ein Assistenzsystem welches den Fahrzeugführer präventive auf bevorstehende Kollisionen hinweist trägt dazu bei, dass der Straßenverkehr sicherer wird. Durch ein Abnehmen von Unfällen sinkt die Versicherungsprämie, die Fahrt ist komfortabler, Stress freier, es motiviert unerfahrene Fahrer an dem Straßenverkehr teilzunehmen und bietet routinierten Fahrern einen zusätzlichen Rückhalt. Ein Assistenzsystem, dass auf dem persönlichen Mobiltelefon des Fahrzeugführers, oder Mitfahrer arbeitet, verursacht keine Anschaffungskosten und kann unabhängig von dem Fahrzeugtyp funktionieren.

Um den Straßenverkehr sicherer zu erleben, muss ein Assistenzsystem eingeführt werden, dass den Fahrzeugführer unterstützt und somit Schäden vorbeugt. In schwer einsehbaren Situationen wie an Ampeln, Seitenstraßen oder bei Nebel und starkem Regen kann ein Assistenzsystem Unfälle vereiteln und somit durch das austauschen von Daten der einzelnen Teilnehmer Menschenleben retten . Ein Auffahrunfall kann sich zu einen schweren Unfall entwickeln mit vielen Beteiligten, wenn kein Assistenzsystem zur Verfügung steht, das einen Nachrichtenaustausch unter mehreren Fahrzeugen herstellt. Ein starkes Bremsen, oder eine ungewöhnliche Situation wird in weniger als einer Sekunde unter allen Verkehrsteilnehmern distributiert, um eine Kollision zu vermeiden.

Viele Unfälle entstehen aus harmlosen Situationen, die erst durch die Fahrlässigkeit des Fahrzeugführers gefährlich werden und somit Dritte in Gefahr bringt. Ein Assistenzsystem, das günstig in der Anschaffung ist und den Fahrer im Verkehr unterstützt, wird den Komfort, Effizienz und Sicherheit auf Straßen anheben. Ein Ortungsdienst auf dem Mobiltelefon aktualisiert durchgehend die Position der Anwender, sodass die Position die Grundlage des Assistenzsystem ist und durch das WLAN Modul zur Verfügung gestellt wird. Dadurch verfügt das Assistenzsystem über alle Positionen und Geschwindigkeiten der im Umkreis liegenden Fahrzeuge, sodass die Bewegungen berechnet und der Anwender

auf gefährliche Situationen frühzeitig hingewiesen wird und auf diese angemessen reagieren kann.

Diese Arbeit soll die Frage klären, wie ein Assistenzsystem aufgebaut ist um im Straßenverkehr Kollisionen zu vermeiden. Im Kapitel 2 werden die Grundlagen erläutert, die für das Verständnis der vorliegenden Arbeit nötig sind. Im Kapitel 3 Verwandte Arbeiten wird ein kurzer Überblick über die Forschungsarbeiten im C2X Bereich gegeben. Gefolgt von dem Kapitel 4 Analyse. Aus dem Ziel der Arbeit und der Ist-Analyse im Kapitel Analyse, sind die Anforderungen und Arbeitspakete für diese Ausarbeitung abgeleitet. Das Kapitel Analyse endet mit der Definition der Hypothesen, die für diese Ausarbeitung zu belegen sind. Anschließend folgt das Kapitel 5 Konzept, welches auf Basis der Anforderungen die Arbeitspakete konzeptionell ausarbeitet, sodass die Implementierung im Kapitel 6 stattfindet. Im Kapitel 7 werden die Simulationsergebnisse in Diagrammen und Tabellen dargestellt, analysiert und interpretiert. Basierend auf der Interpretation der Diagramme sind die Hypothesen im Abschnitt 4.7 zu belegen, sodass basierend auf den Forschungsergebnissen im Kapitel 8 eine Zusammenfassung, Fazit und Ausblick gegeben wird.

2 Grundlagen

In diesem Kapitel werden die Grundlagen vermittelt, die für das Verständnis dieser Arbeit notwendig sind. Im Kapitel 2.1 werden die 7 Layer des OSI-Modells erläutert und im Kapitel 2.2 eine Auswahl von Protokollen für Layer 1-4 im OSI-Modell vorgestellt. Das Kapitel 2.3 stellt kurz den Standard 802.11p vor und folgend die technischen Möglichkeiten von Bluetooth. Im Kapitel 2.5 sind die Grundlagen von der Simulationsumgebung OMNeT++ erklärt. Im Kapitel 2.6 wird der Verkehrssimulator SUMO vorgestellt zusammen mit dem Car Following Models und Vehicle Type Definitionen. Im Kapitel 2.7 wird Veins vorgestellt und zwei Modelle die dazu beitragen das Übertragungsmedium realitätstreuer zu modellieren. Das Kapitel 2.8 erläutert wie SUMO mit Veins gekoppelt wird um Zugriff auf Simulationsobjekte zu erlangen und das simulierte Verhalten in SUMO zu beeinflussen. INET wird im Kapitel 2.9 vorgestellt und anschließend im Kapitel 2.10 wird erläutert, was unter einem Random Number Generator zu verstehen ist. Abschließend werden im Kapitel 2.11 zwei Methoden zur Positionsbestimmung vorgestellt, unterschiedliche Methoden zur Berechnung der Position, Mozilla Location Service und das Global Positioning System.

2.1 OSI-Modell

Das OSI-Modell (Open System Interconnection) wurde für die Kommunikation in offenen Systemen entwickelt[47]. Das OSI-Modell ist hierachisch in sieben Layer organisiert, wobei jede Schicht unabhängig von allen anderen Dienste zur Verfügung stellt[47] die den nächst höheren Layer unterstützen. Die Schichten 1-4 werden als Transportorientierte und 5-7 als Anwendungsorientierte Schichten bezeichnet[47].

1. **Physical Layer**

   Der Physical Layer definiert die elektrische, mechanische und funktionale Schnittstelle zum Übertragungsmedium[50].

2. **Data Link Layer**

Der Data Link Layer soll Funktionen definieren für die Fehlererkennung, die in Layer 1 eintreten können, Fehlerbehebung und Datenflusskontrolle[47].

3. **Network Layer**

   Der Network Layer koordiniert die logische Trennung der Kommunikation von zwei Gesprächspartnern, die logisch adressiert sind[50].

4. **Transport Layer**

   Im Transport Layer können verbindungslose und verbindungsorientierte Transporttechniken implementiert werden, die Algorithmen für Fluß- und Staukontrolle beinhalten können[47]. Die Staukontrolle ist eine Reaktion auf eine Überlast auf dem Übertragungsmedium und die Flußkontrolle eine Laststeuerung durch den Empfänger.

5. **Session Layer**

   Der Session Layer definiert die Verbindung zwischen zwei Kommunikationspartnern.

6. **Presentation Layer**

   Der Presentation Layer übersetzt die Daten in das bevorzugte Format für die Anwendungsorientierte bzw. Transportorientierte Schicht[47].

7. **Application Layer**

   Der Application Layer kann Dienste wie die Datenübertragung, Fernsteuerung von Computern und Namensauflösung implementieren.

2.2 Protokolle

In diesem Kapitel wird für die Layer 1-4 im OSI-Modell eine Auswahl von Protokollen vorgestellt.

2.2.1 Physikalischer Layer

Das Orthogonal Frequency-Division Multiplexing (OFDM) Modulationsverfahren überträgt mehrere Datenströme auf unterschiedlichen Trägerfrequenzen. OFDM ist eine Sonderform von FDM, durch Orthogonalität der Trägerfrequenzen wird ein übersprechen der Signale minimiert. Das Datensignal wird im Trägersignal durch Änderung der Amplitude oder Phasenwinkel dargestellt.

2.2.2 MAC Layer

Der Media Access Control (MAC) Layer definiert die priorisierte Zuteilung und Zugriff auf das Übertragungsmedium. Folgend sind die Protokolle EDCA, CSMA/CA, TDMA und STDMA vorgestellt.

EDCA

Im Enhanced Distributed Channel Access (EDCA) Protokoll wird ein priorisierte Zuteilung des Übertragungsmediums definiert durch Einführung von Channel Categories (AC), wobei AC0 die höchste und AC3 die niedrigste Priotät hat. Dies wird in der Abbildung 2.1 dargestellt.

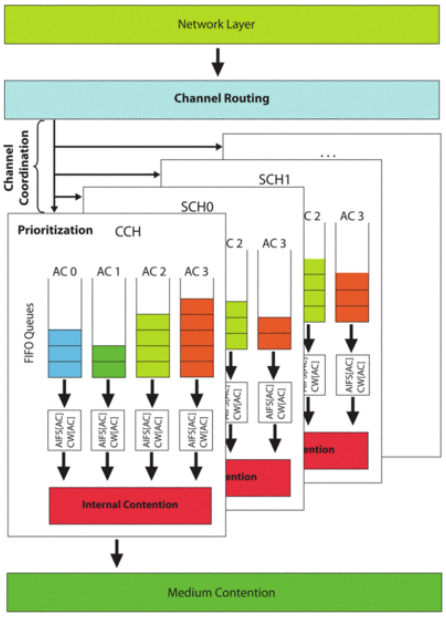


Abbildung 2.1: Beziehung zwischen den Kanälen und der Traffic-Priorisierung durch unterschiedliche Channel Access Gruppen im 802.11p MAC[48].

Die Abbildung 2.1 zeigt, wie im ersten Schritt eine Kanalzuordnung stattfindet (*Channel Coordination*) und in jeden Kanal eine weitere Zuordnung nach Priorität (vgl. AC0 - AC3). Jeder AC wird ein *Arbitration Inter-Frame Space* (AIFS)[49, S. 13] Timer zugewiesen, welche die Priorität vom AC darstellt (vgl. *AIFS[AC]* und *CW[AC]*, siehe Abbildung 2.1). Alsbald ein Paket in Abhängigkeit seiner AC priorisiert ist findet die Zuteilung vom Übertrsagungsmedium statt. AC3 ist AC2 gegenüber minimal, AC1 und AC0 annäherungsweise um Faktor 2 bzw. 3 priorisiert[48, Abbildung 4].

Carrier Sense Multiple Access

Carrier Sense Multiple Access (CSMA) ist ein dezentrales und asynchrones Verfahren für den Zugriff auf ein Übertragungsmedium. Jeder Sender muss vor dem Senden das Übertragungsmedium abhören und wenn das Übertragungsmedium für ein definiertes Zeitinterval (AIFS) frei ist, wird die Übertragung gestartet[52]. Es wird in drei Arten von CSMA unterschieden

1. **1-Persitent**

   Wenn das Medium frei ist wird immer gesendet.

2. **P-Persistent**

   Es wird mit einer Wahrscheinlichkeit von P gesendet, wenn der Kanal frei ist.

3. **Non-Persistent**

   Wenn der Kanal besetzt ist wird eine Zufällige Zeit gewartet, die sich aus dem Backoff Algorithmus ergibt, anschließend wird der Kanal erneut überprüft und gesendet, wenn der Kanal frei ist.

Carrier Sense Multiple Access/Collision Avoidance

CSMA/CA[50] ist eine Non-Persistente Erweiterung vom CSMA Protokoll. Die Collision Avoidance (CA) wird durch den ***Binary Exponential Backoff*** Algorithmus erreicht. Durch die Formel $2^{Sendeversuche}$ wird zufällig ausgewählt in welchen Slot $[0, 2^{Sendeversuche} - 1]$ der nächste Sendeversuch angestrebt wird. Nach 16 Senderversuchen wird der Algorithmus abgebrochen, wenn der Physikalische Layer OFDM ist[51, S. 303].

Time Division Mulitple Access

Time Division Mulitple Access (TDMA)[53] teil den Kanal in Time Slots auf die von unterschiedlichen Sendern genutzt werden, wobei in ein Synchrones und Asynchrones Verfahren unterschieden wird[53]. Bei dem Synchronen Verfahren wird jedem Sender ein Time Slot auf dem Übertragungsmedium zugeordnet, wohingegen bei dem Asynchronen Verfahren nicht belegte Time Slots nach Bedarf in Anspruch genommen werden, wenn der Slot unbenutzt ist[53].

Self-Organizing Time Division Multiple Access

Self-Organizing Time Division Multiple Access (STMA)[52] ist ein MAC Protokoll mit einen Algorithmus für die Zuteilung von Slots für mehrere Sender im selben Kanal, auf Basis der Positionen der einzelnen Sender. Der im STMA verwendete Algorithmus wurde von Hflkan Lans entwickelt[52] und ist in vier Schritten definiert.

1. Jeder Teilnehmer sendet seine Position periodisch, sodass in der Initial Phase jeder Teilnehmer für einen Frame das Übertragungsmedium abhört um die Anzahl der Teilnehmer und die angeforderte Anzahl der Slots je Sender (Report Rate) festzustellen[52].

2. In der Network Phase bestimmt jeder Teilnehmer seine Slots durch folgenden Algorithmus:

   a) Bestimme den Nominal Increment $NI = \frac{Number\ of\ Slots}{Report\ Rate}$[52].

   b) Wähle zufälligen Nominal Start Slot (NSS) innerhalb des Intervalls [aktueller Slot, NI][52].

   c) Bestimme $SI = 0,2 * NI$ und merke SI Slots um NSS vor[52].

   d) Wähle zufällig den nominal Transmission Slot (NTS) innerhalb von SI. Wenn der ausgewählte NTS vergeben ist, dann wähle den nähesten Slot innerhalb von SI. Wenn alle Slots im SI belegt sind, dann wähle den Slot, wessen Sender am weitesten entfernt ist[52].

3. In der First Frame Phase bestimmt der Nominal Slot (NS) den nächsten Slot im selben Frame, wobei das vorgehen wie unter Punkt zwei ist. Der Punkt zwei wird solange wiederholt bis in Abhängigkeit von der Report Rate jeder Sender im Frame die angeforderte Anzahl an Slots belegt[52].

4. In der Continuous Phase sendet jeder Teilnehmer in Abhängigkeit der belegten NTS[52].

Außerdem wählt jeder Teilnehmer für jeden belegten NTS ein $n \in \{3, ..., 8\}$ aus und nach n Durchläufen wird ein neuer NTS im selben SI ausgewählt[52]. In der Abbildung 2.2 wird der STMA Algorithmus in der First Frame Phase dargestellt

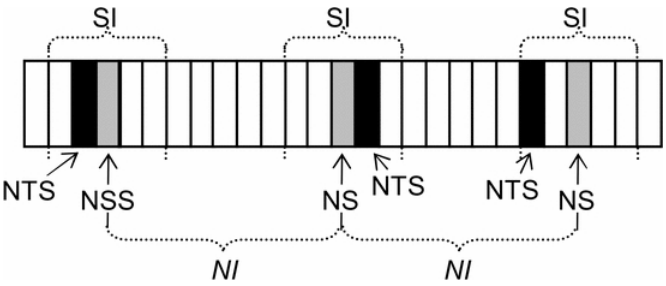


Abbildung 2.2: Darstellung des STMA Algorithmus in der First Frame Phase aus der Sicht eines Senders[52].

2.2.3 Network Layer

Internet Prokoll

Das Internet Protokoll (IP) ist ein verbindungsloses Netzwerkprotokoll das im OSI-Modell auf der Vermittlungsschicht arbeitet. Computer werden durch IP-Adresse und Subnetzmaske in Subnetzwerken organisiert[1]. Die Subnetzmaske gibt an wie viele Bits der IP-Adresse als Netzpräfix zu interpretieren ist, der bei allen IP-Adressen des selben Subnetzes gleich sind. Durch die restlichen Bits der IP-Adresse kann jeder Computer des Netzwerkes eindeutig identifiziert werden. Mit der Ausnahme von speziellen IP-Adressen die reserviert sind für andere Zwecke. Dazu gehören IP-Broadcast Adressen, die von jedem im Netzwerk empfangen werden. Jeder Empfänger kann die Zuständigkeit frei entscheiden und den Broadcast bearbeiten oder verwerfen.

2.2.4 Transport Layer

Transmission Control Protokoll

Das Transmission Control Protokoll (TCP) ist ein Netzwerkprotokoll, das Verbindungsorientiert und zuverlässig ist. Es definiert die Art und Weise wie Computer in Netzwerken miteinander Kommunizieren, indem es die Übertragung überwacht und im Fall von Paketverlusten eine erneute Sendung vornimmt. Dazu existieren unterschiedliche Staukontrollen (Congestion Control) um ein Überlasten des Netzwerkes entgegen zu wirken[2]. Pakete können im Netzwerk von z.B. Routern verworfen werden, wenn die Last an dem Knotenpunkt zu hoch ist -der First in First out (FiFo) Speicher ist voll-, dieses Vorgehen veranlasst den Sender sein Paket erneut zu senden, was in einer höheren Belastung

[1] https://en.wikipedia.org/wiki/IP_address
[2] https://en.wikipedia.org/wiki/Transmission_Control_Protocol

resultiert. Wenn ein Sender ein Paket versendet und innerhalb eines Time Outs keine Bestätigung (Acknowledge) vom Empfänger erhält, ist das Paket verloren und es muss erneut gesendet werden.

Congestion Control

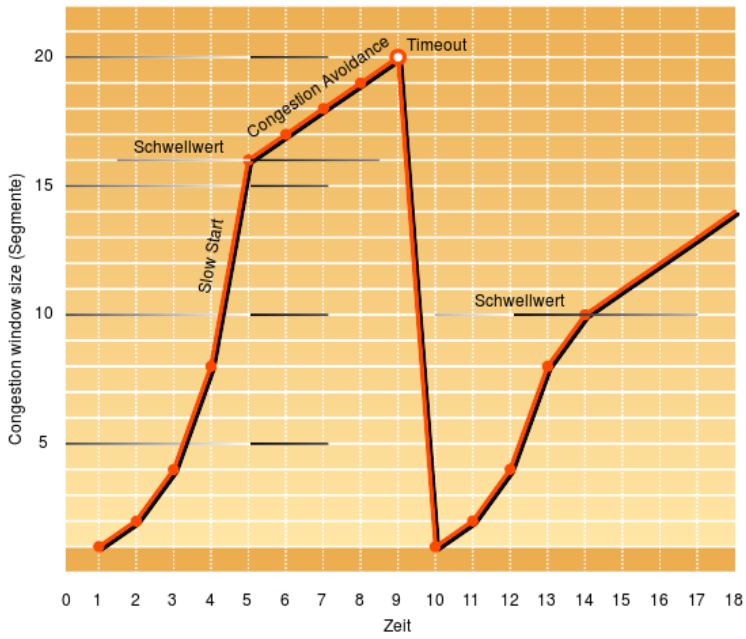


Abbildung 2.3: TCP Slow Start Algorithmus[3].

In Abbildung 2.3 ist der Slow Start Algorithmus dargestellt, der mit einer Congestion Window Size (CWS) von 1 startet. Die maximale Größe des CWS in Bytes wird mit jeder erfolgreichen Übertragung verdoppelt, sodass sich ein exponentielles Wachstum einstellt. Das exponentielle Wachsen des CWS wird gestoppt sobald der Slow-Start-Threshold erreicht ist, welches $\frac{CWS}{2}$ des letzten Paketverlustes entspricht und wächst nur noch linear in einer Schritten und wird Congestion Control genannt. Wenn ein Time Out auftritt wird der Slow Start neu gestartet mit CWS = 1 (vgl. Abb. 2.3). Dieses Verfahren wird durch die Techniken Fast-Retransmit und Fast-Recovery (TCP Reno) verbessert indem Duplicate Acknowledge (ACK) eingeführt werden. Wenn bei dem Empfänger Pakete in der falschen Reihenfolge ankommen, wird für das zuletzt in

[3] https://de.wikipedia.org/wiki/Datei:TCPSlowStartundCongestionAvoidance.svg

richtiger Reihenfolge angekommene Paket, ein ACK für jedes weitere Paket gesendet, sodass der Sender erkennt welches Paket nicht angekommen ist und sendet nach dem dritten ACK, vor dem Ablaufen des Timer Outs, das fehlende Paket (Fast-Retransmit). Nach dem Fast-Retransmit wird das CWS halbiert (Fast-Recovery).

User Datagram Protokoll

Das User Datagram Protokoll (UDP) ist ein verbindungsloses Netzwerkprotokoll das Nachrichten in einem Datagramm mit Nutzlast, Zielport und Prüfsumme - zur Fehlererkennung - versendet und ermöglicht den Datenaustausch von Prozess zu Prozess. UDP stellt hierbei jedoch nicht sicher, ob ein Paket in der richtigen Reihenfolge versendet wird, ob es überhaupt- oder mehrmals ankommt. So handelt es sich bei UDP lediglich um ein verbindungsloses Protokoll das einen schnellen Datenaustausch ermöglicht.

2.3 IEEE 802.11p

Die Protokolle (EDCA), (CSMA/CA) im MAC Layer und die Modulationstechnik OFDM im Physikalischen Layer sind zusammen von der IEEE als 802.11p standardisiert[62]. Der Standard P soll den Ansprüchen von kommunizierenden Verkehrsteilnehmern gerecht werden und definiert 10Mhz breite Kanäle innerhalb des 75Mhz Frequenzbandes im 5,9Ghz Band[62]. Es sind sechs Service und ein Control Kanal zu je 10Mhz Bandbreite definiert (vgl. Abbildung 2.4).

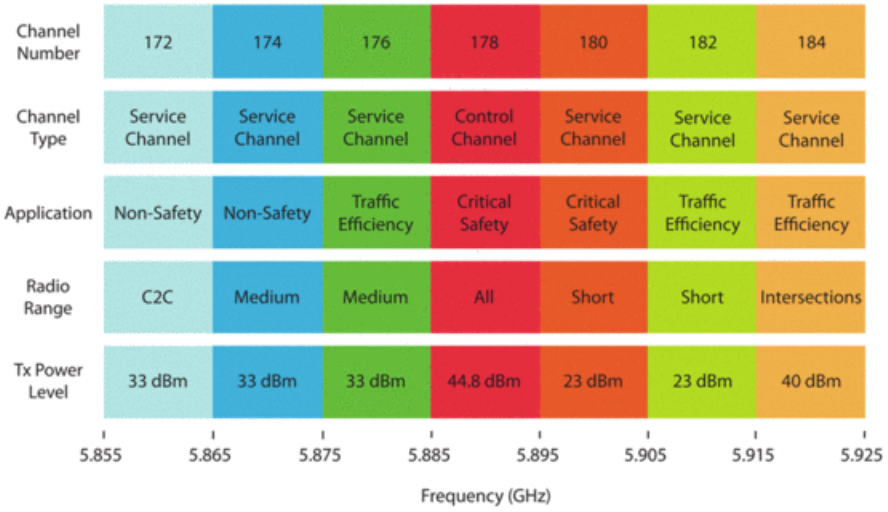


Abbildung 2.4: Kanaldefinition für den 802.11p Standard[48].

In Abbildung 2.4 ist die Aufteilung der 75Mhz Bandbreite in sieben Kanäle aufgezeigt und den Verwendungszweck notiert als *Application*. Fahrzeugnetzwerke sind dezentralisierte Netzwerke, sodass keine Frequenzzuteilung durch einen zentralen Sender an die einzelnen Verkehrsteilnehmer statt findet (vgl. [15]) und innerhalb des Standards definiert ist, welcher Frequenzbereich für welche Art von Anwendung zu nutzen ist.

2.4 Bluetooth

Der neuste Bluetooth Standard 4.0 (Class 2) (IEEE 802.15.1), besitzt einen Datendurchsatz von 24 $\frac{Mb}{s}$ und verbindet bis zu 10m entfernte Endgeräte miteinander[17].

2.5 OMNeT++

Objective Modular Network Testbed in C++ (OMNeT++)[44] ist eine Objektorientierte, Modulare und ereignisdiskrete Simulationsumgebung für die Modellierung von Kommunikationsnetzwerken, Protokollen und verteilten Systemen[45].

2.5.1 Struktur

Simulationen in OMNeT++ bestehen aus Modulen, die untereinander durch Message Passing kommunizieren[45, Kapitel 2.1]. Es wird in Simple und Compound Modulen unterschieden, wobei ein Compound Module Untermodule enthält, sodass die Hierachie unbegrenzt ist[45, Kapitel 2.1]. Die höchste Hierachiestufe bzw. das gesamte Modell wird als Network bezeichnet[45, Kapitel 2.1]. Die Modell Struktur ist durch die Network Description Language (NED) beschrieben[45, Kapitel 3.1] und die Logik der Komponenten durch C++. In Abbildung 2.5 ist ein Netzwerk zu sehen, das aus drei Simple und einem Compound Modul besteht. Simple Moduls besitzen Gates -in der Abbildung als Quadrat an den Modulen dargestellt- die durch Connections -in der Abbildung als Pfeile dargestellt- miteinander verbunden sind. Eine Kommunikation durch Nachrichten unter Modulen kann direkt oder über Connections stattfinden. Dazu hat jedes Modul eine handleMessage() Methode zum Empfangen von ankommenden Nachrichten.

2.5.2 NED Sprache

Die Network Description (NED)[45, Kapitel 3.1] Sprache beschreibt die Struktur des Simulationsmodells. Mit NED wird definiert aus welchen Modulen das Network besteht und wie die einzelnen Komponenten miteinander verbunden sind[45, Kapitel 3.1].

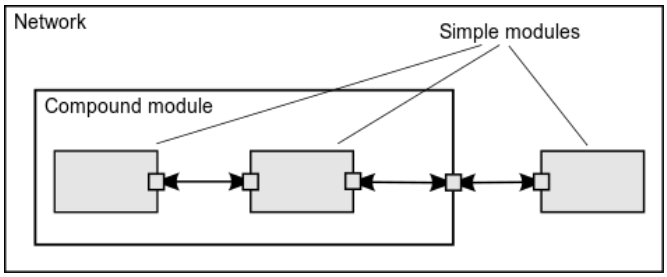


Abbildung 2.5: Darstellung von Simple Modules, Compound Modules und Network in OMNeT++. Durch die Gates wird ein Datenaustausch unter den Modulen ermöglicht, die in der Abbildung als Quadrate an den Modulen dargestellt sind[45, Kapitel 2.1].

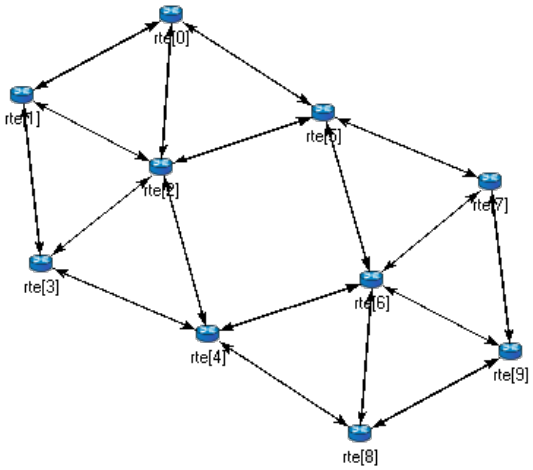


Abbildung 2.6: Grafische Darstellung einer Network Topologie, die durch die NED Sprache definiert ist[45, Kapitel 3.2.1].

Parameter können aus der NED-Datei in die Module des Networks übergeben werden, sodass eine Parameterisierung aus höchsten Hierachieebene möglich ist[45, Kapitel 3.6]. In der Abbildung 2.6 ist eine grafische Darstellung einer Network Topologie zu sehen, bei der, durch die NED Sprache, die Komponenten rte[0] - rte[9] miteinander verbunden sind. Die Verbindungen sind dargestellt durch Connections, denen im NED-File eine Übertragungsgeschwindigkeit zugewiesen wird.

2.5.3 Module und Connections

OMNeT++ Simulationsmodelle bestehen aus Simple Modulen und Connections, wobei die Simple Module die aktiven Komponenten des Modells sind und ihr Verhalten durch den Anwender in C++ definiert ist[45, Kapitel 4.2]. Connections können Channels sein, deren Verhalten auch durch den Anwender in C++ definiert ist. Mithilfe von Channels können Übertragungszeiten, Übertragungsfehler und Übertragungsverzögerungen definiert werden. Die Abbildung 2.7 zeigt das UML-Diagramm und die Verbindung der Klassen *cSimpleModule* und *cChannel*.

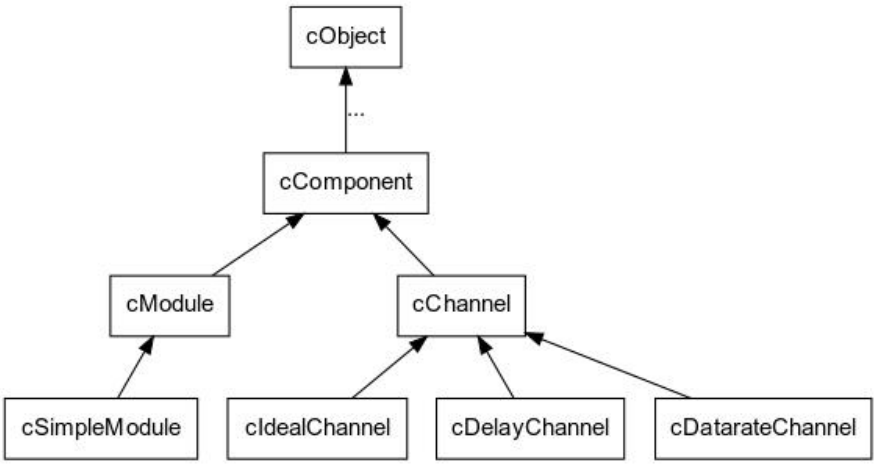


Abbildung 2.7: UML-Diagramm der Klassen cComponent, cModule und cChannel[45, Kapitel 4.2].

Simple Module und Channels werden durch Neudefinition einer Menge von Funktionen programmiert, indem der Anwender in der folgenden Auswahl der wichtigsten Funktionen, die gewünschte Funktionalität implementiert[45, Kapitel 4.2].

1. **initialize()**
   - OMNeT++ ruft die initialize() Methode auf, nachdem das Network Konfiguriert ist. Die initialize() Methode bietet Platz für Initialiserungen[45, Kapitel 4.2].
2. **finish()**
   - OMNeT++ ruft die finish() Methode auf, wenn die Simulation erfolgreich beendet ist. Die finish() Methode ist der Empfohlene Bereich zur Erfassung der Statistiken[45, Kapitel 4.2], außerdem können (Message) Pointer freigegeben werden.
3. **handleMessage(cMessage *msg)**
   - Diese Funktion wird aufgerufen, wenn das Modul eine Nachricht empfängt. Außerdem verstreicht keine Simulationszeit in handleMessage()[45, Kapitel 4.2].

2.5.4 Nachrichten und Pakete

Nachrichten sind ein zentrale Konzept in OMNeT++. Nachrichten repräsentieren Ereignisse, Pakete, Befehle und Aufgaben (vgl. [45, Kapitel 5.1]). Nachrichten sind durch die *cMessage* Klasse repräsentiert und ihrer Unterklasse *cPacket*, welche für Netzwerkpakete genutzt wird wie Ethernet Frames, UDP Datagramme, IP Pakete und cMessage für alle anderen Zwecke. Im folgenden sind die wichtigsten Parameter von *cMessage* zusammengefasst.

1. **Message Name**
   - Die Variable *Name* beinhaltet den Namen der Nachricht, welcher in grafischen Darstellungen repräsentative für die jeweilige Nachricht steht.
2. **Message Kind**
   - Message *Kind* ist eine Integer Variable, die den Typ, die Rolle, Kategorie oder Identität der Nachricht bestimmt[45, Kapitel 5.1]. Dies kann z.B. durch ein Aufzählungstyp (Enum) erreicht werden.

Die Klasse cPacket ist eine Unterklasse von cMessage und enthält zusätzlich folgende wichtige Variablen.

1. **Packet Length**

- Packet Length repräsentiert die Größe des Packets in Bits und wird vom Simulationskernel genutzt um die Übertragungszeit zu ermitteln, bei einer Connection mit einer bestimmten Übertragungsgeschwindigkeit und für die Modellierung von Errors, wenn die Connection eine $ErrorRate > 0$ aufweist[45, Kapitel 5.1].

2. **Bit Error Flag**
   - Das Feld Bit Error Flag beinhaltet das Ergebnis der Error Modellierung, nachdem das Packet über ein Channel gesendet ist mit einer $PacketErrorRate > 0$ und $BitErrorRate > 0$[45, Kapitel 5.1].

Self-Messages

Simulationen in OMNeT++ bestehen häufig aus wiederkehrenden Aufgaben in einen wohldefinierten Zeitinterval, wie dem Ausführen einer bestimmten Methode. Diese Time-Outs werden durch Self-Messages implementiert. Self-Messages sind *cMessages*, die Time-Outs für bestimmte Aufgaben innerhalb eines Moduls implementieren und durch die Methode *scheduleAt()* zu einer zukünftigen Simulationszeit als Ereignis eingeplant werden. Die Methode *isSchedule()* stellt fest, ob eine Self-Message eingeplant ist und mit *cancelEvent()* wird das Ereignis, das eine geplante Self-Message auslöst, gelöscht. Nachrichten treffen im Modul, in der Methode *handleMessage()* ein und sind durch die Variable *kind* unterscheidbar. Anschließend wird die zugehörige Methode ausgeführt und der Timer durch *scheduleAt()* neu gestellt.

2.5.5 Ergebniserfassung

Die Ergebniserfassung wird in OMNeT++ durch Output Vectors und Output Scalars durchgeführt. Vectoren sind Daten mit Zeitstempel, erfasst durch ein Simple Modul oder Channel. Scalar Daten werden während der Simulation berechnet und nach dem Simulationsende gespeichert in Form einer ganzen oder reelen Zahl. Ergebnisse können auf zwei Wegen erfasst und gespeichert werden:

1. Auf Basis von Signalen, durch angemeldete Statistiken.
2. Direkt vom C++ Quellcode, durch die Simulationsbibliothek.

Die zweite Methode ist der traditionelle Weg, während der erste Weg mit OMNeT 4.1 eingeführt ist und bevorzugt ist, weil das Generieren von Ergebnissen entkoppelt von

der Ergebniserfassung ist[45, Kapitel 12]. Signale müssen in den OMNeT++ NED-File registriert werden um zur Datenerfassung genutzt zu werden. Das wird erreicht durch das Listing 2.1 in der initialize() Methode.

```
1 simsignal_t signal = registerSignal(signalName);
```

Listing 2.1: Registrierung eines Signals für die statistische Datenerfassung.

Anschließend können Signale ausgesendet werden durch die Funktion im Listing 2.2.

```
1 emit(signalName, &signalValue);
```

Listing 2.2: Ausgesendete Signale mit zugehörigen Wert für die statistische Auswertung.

Durch *@statistic[nameOfStatistic]* werden Signale als *nameOfStatistic* im zugehörigen NED-File registriert. Es ist möglich Berechnungen auf den empfangenen Signalen durchzuführen, bevor diese in die Scalar bzw. Vector Datei geschrieben werden (vgl. Listing 2.3), sodass in *Result Filters* und *Result Recorder* unterschieden wird, wie die Abbildung 2.8 zeigt.

```
1 @statistic[droppedBits](source=8*packetBytes(pkdrop); record=sum,
    vector(sum));
```

Listing 2.3: Durch Result Filter können Signale vor dem speichern manipuliert werden[45, Kapitel 4.15.2.5].

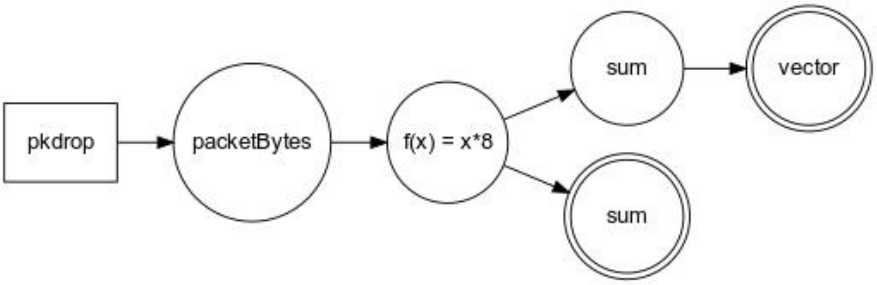


Abbildung 2.8: Result Filter und Recorder in einer Beispielkette[45, Kapitel 4.15.2.5].

In Listing 2.3 und in der zugehörigen Abbildung 2.8 sind drei Result Filter und zwei Result Recorder zu erkennen. Der erste Result Filter ist *packetBytes()*[4] der den Pointer *pkdrop* in Bytes umwandelt, in Abhängigkeit von der Größe des Pakets auf das zeigt wird. Im Anschluss wird das Ergebnis im zweiten Filter mit 8 multipliziert, um die Größe in Bits zu erhalten. Der erste Result Recorder ist das *sum* im Doppelkreis. In dem

[4] packetBytes() erwartet einen Pointer vom Typ *cPacket*[45, Kapitel 4.15.2.5].

Recorder wird die Summe über alle empfangenen Werte gebildet und in eine Scalar Datei geschrieben, nachdem die Simulation beendet ist. Der Obere Arme mit dem Filter *sum* berechnet die Summe aller bisherigen Werte und gibt den aktuellen Wert im Anschluss an den Recorder Vector, welcher die Paare *(Wert und Simulationszeit)* in eine Vector Datei schreibt. Aus der Abbildung 2.8 ist weiterhin zu entnehmen, dass Funktionen wie sum, aber auch min, max, count und weitere sowohl Result Filter als auch Recorder sind (vgl. [45, Kapitel 4.15.2.5]).

2.6 SUMO

Simulation of Urban Mobility (SUMO) ist ein Open Source Verkehrssimulator. SUMO simuliert wie sich Verkehr, welcher aus einzelnen Fahrzeugen besteht, durch ein Straßennetzwerk bewegt. SUMO unterstützt die Modellierung jedes Fahrzeuges im Netzwerk mit individueller Route und individueller Bewegung. SUMO Simulationen sind deterministisch, dennoch existieren Optionen mit der zufälliges Verhalten auf die Simulation Einfluß hat[7]. Folgend wird das Car Following Modell vorgestellt und die Vehicle Type Beschreibung, in der Automobile und ihre physikalischen Eigenschaften definiert sind, sowie zu fahrende Route und zufälliges Verhalten.

2.6.1 Car Following Modell

SUMO stellt unterschiedliche Car-Following Modelle (CFM) bereit, die das Verhalten des Fahrers bzw. Fahrzeug im Straßenverkehr beschreibt. Die Modelle unterscheiden sich in den Inizialisierungswerten der Attribute Tabelle 2.1 und die darauf ausgeführten Berechnungen um das Verhalten der Automobile zu bestimmen.

Attribute	Beschreibung
Accel	Max. Beschleunigung in $\frac{m}{s^2}$
Decel	Max. Bremsung in $\frac{m}{s^2}$
Sigma	Fehleranfälligkeit des Fahrers zwischen 0 und 100%
Tau	Reaktionsgeschwindigkeit in Sekunden
MinGab	Mindestabstand in Metern
Security	Wunsch nach Sicherheit
Estimation	Genauigkeit der Einschätzung der Situation

Tabelle 2.1: Auswahl der Attribute in Car Following Modellen, die das Verhalten von Verkehrsteilnehmern bestimmen[7].

Das *Krauss Modell* ist das Standard CFM in SUMO[26] und ist ein Kollisionsfreies Modell[27]. Automobile mit Krauss CFM fahren so schnell wie möglich, solange die Sicherheit der Fahrzeug nicht beeinträchtig ist, innerhalb der Beschleunigungsgrenzen der Verkehrsteilnehmer[9].

2.6.2 Vehicle Type Beschreibung

In der Vehicle Type Beschreibung sind die Automobile, ihre physikalischen Eigenschaften, zu fahrende Route und CFM Parameter für die Simulation in SUMO definiert (vgl. Tabelle 2.1, [9]).

```
1 <routes>
2    <vType id="vtype0" accel="8.0" decel="8.0" carFollowModel="
        Krauss" speedFactor="1" speedDev="0.1" length="6.0" minGap=
        "0" maxSpeed="14" color="1,1,0"/>
3
4    <route id="route0" edges="-3998643#1␣-3998643#0"/>
5
6    <flow id="flow0" type="vtype0" route="route1" begin="0" period
        ="14" number="10"/>
7 </routes>
```

Listing 2.4: Vehicle Type und Flow Definition innerhalb der Route Definition für die SUMO Simulation.

Im Listing 2.4 ist eine vollständige Vehicle Beschreibung zu sehen, eingebettet in einem *<routes>* Tag. Im *vType* Tag sind die physikalischen Eigenschaften festgelegt, CFM und

zugehörige Parameter und der Name für den definierten Fahrzeugtyp. Der Fahrzeugtyp *vtype0* ist definiert durch eine Beschleunigung von $\pm\frac{8m}{s^2}$, Länge von 6m, maximale Geschwindigkeit von $\frac{14m}{s}$, Mindestabstand von 0m zum vorausfahrenden Fahrzeug und die Fahrzeugfarbe mit dem RGB Code von (1,1,0). Die maximale Geschwindigkeit in der Simulation wird durch die Variablen *speedFactor* und *speedDev* bestimmt[42]. Mit dem Erwartungswert als speedFactor und der Standardabweichung als speedDev wird eine maximale Geschwindigkeit implementiert, bei der 95% der Fahrzeuge zwischen 80% und 120% der erlaubten Geschwindigkeit fahren, indem die maximale Geschwindigkeit der Straße, auf der sich das Fahrzeug befindet, mit dem Wert aus der Verteilung multipliziert wird[42]. Durch die Verteilung der maximalen Geschwindigkeit ist das Verhalten der Fahrzeuge indeterministischer.

Durch das Einbetten der *vType* Definition im *<routes>* Tag kann folgend eine Route und Flow definiert werden. Eine Route ist eine Festlegung von Straßennamen, die sequenziell abgefahren werden. Im Listing 2.4 wird durch den Tag *route*, *edges=* und zwei Straßennamen die Route *route0* definiert.

Ein Flow definiert wann (Simulationszeit, *begin*), in welchem Intervall (*period*), was für Fahrzeuge (*vtype0*), wieviele Fahrzeuge (*number*) in die Simulation eingefügt werden und welche Route (*route0*) abgefahren wird. Durch den definierten Flow *flow0* kann der Verkehr in einer Simulation gesteuert werden.

2.7 Veins

Veins ist ein Simulationsframework für die Simulation von Netzwerken bestehend aus Automobilen. Veins stellt Modelle für die Vernetzung von Fahrzeugen bereit, die von dem Ereignis basierten Netzwerksimulator OMNeT++ (siehe 2.5) ausgeführt werden. Parallel kommuniziert Veins mit dem Road Traffic Simulator Simulation of Urban Mobility (SUMO, siehe 2.6, vgl. [46]). Veins dient als Ausführungsumgebung für Simulationsmodelle wie Car-to-Car (C2C) bzw. Car-to-X[5] (C2X) Anwendungen und ist verantwortlich für das modellieren von Übertragungsmedium, Physical und MAC Layer und Fahrzeug Mobilität (Position und Bewegung)[3]. Veins stellt für den Physical und MAC Layer die IEEE 802.11p implementierung bereit, das für den Anwendungsfall von C2X Anwendungen Standartisiert ist.

Die Abbildung 2.9 zeigt den Simulationsaufbau von OMNeT++ als Netzwerksimulator, Veins als Simulator für Netzwerke bestehend aus Fahrzeugen und SUMO (siehe 2.6) als

[5] Ein-Fahrzeug-zu-vielen-Fahrzeugen Relation.

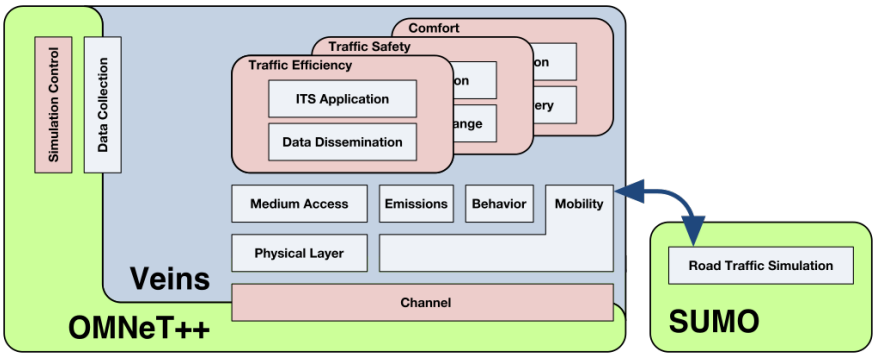


Abbildung 2.9: Veins Framework Topologie innerhalb von OMNeT++, zusammen mit TraCI als Pfeil zwischen Veins und SUMO[3].

Straßenverkehrssimulator. Veins und SUMO sind durch ein TCP Socket miteinander verbunden, sodass eine bidirektional verbundene Simulation von Netzwerk und Straßenverkehr möglich ist[3]. Automobile in SUMO sind durch Nodes in der OMNeT++ Simulation dargestellt und können das Verhalten der Fahrzeuge innerhalb der SUMO Simulation beeinflussen[3]. Veins ist verantwortlich für die Modellierung des Protokollstacks im Layer 1 und 2, was durch IEEE 802.11p erreicht wird, und die Modellierung des Übertragungsmediums. Folgend sind zwei Modelle vorgestellt die Veins implementiert, um das Übertragungsmedium zu modellieren. Signale von Automobilen und Häusern reflektieren, überlagern und beeinträchtigen sich gegenseitig. Dies wird von dem Modell im Kapitel Two-Ray Interference Modell 2.7.1 implementiert. Eine generelle Abschwächung der Signale durch Hindernisse, Mauern und Hauswänden wird durch das Obstacle Shadowing Modell im Kapitel 2.7.1 implementiert.

2.7.1 Physikalische Modelle

Two-Ray Interference Modell

Durch das Two-Ray Interference Modell (2RIM) wird die Überlagerung von Signalen und die daraus resultierende Beeinträchtigung modelliert, welche durch Reflexionen von Automobilen, Häusern, dem Boden oder anderen Gegenständen auftreten in einem Urbanen Umfeld[54, 55]. 2RIM stellt eine Verbesserung von dem Two-Ray Ground Modell (2RGM) dar, welches davon ausgeht, dass eine Signalabschwächung ausschließlich über die Distanz Eintritt[54, 55]. Diese Verbesserung zeigt Abbildung 2.10 auf, in der zum einen reale Messwerte aufgetragen sind und zum anderen die modellierten Werte von 2RIM und 2RGM.

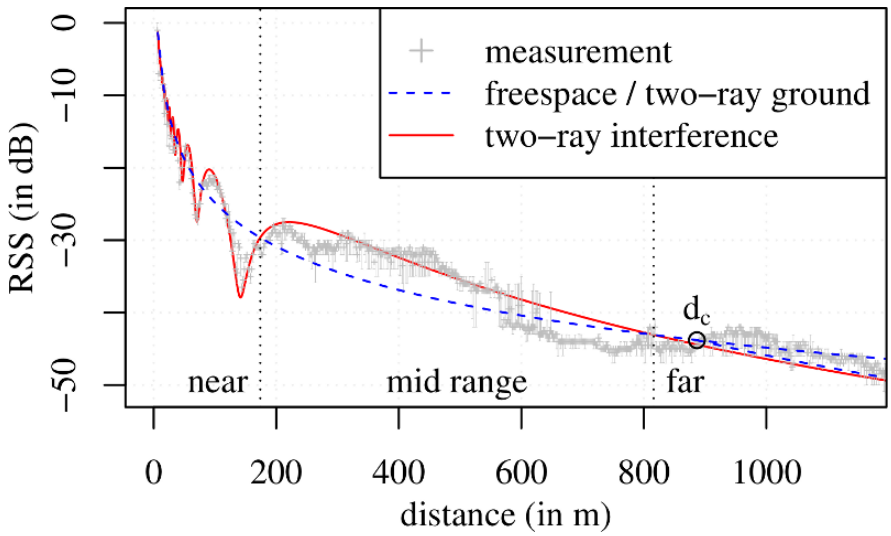


Abbildung 2.10: Vergleich der Interference Modelle Two-Ray Ground und Two-Ray Interefence relative zu realen Messwerten[55]. Mit $d_c < 866,6m$[55, S.2].

Der Abbildung 2.10 ist zu entnehmen, dass das 2RIM Modell relative zu den Messwerten, die Beeinträchtigungen abbildet und auf kurzer Distanz, verglichen mit 2RGM, ein vergleichbares Verhalten zu der Realität liefert (vgl. [54, S. 68][55, S. 3]).

Obstacle Shadowing Modell

Durch das Obstacle Shadowing Modell (OSM) ist die Signalblockierung durch Häuser und Mauern modelliert, sowie die Signalabschweichung in Blicklinie[5, S. 85,87]. In Abbildung 2.11 ist die Signalabschwächung in Blicklinie (dargestellt im gelb-orangen Farbverlauf) und die Signalblockierung (dargestellt in roter Farbe), sowie in grüner Farbe den Signalverlauf der keiner Blockierung unterliegt.

Die Abbildung 2.11 zeigt, dass OSM für die Simulation von Inter-Vehicle Communication (IVC) wichtig ist, da es eine Schätzung über die Signaldämpfung macht, die durch Häuser oder Mauern hervorgerufen werden[5, S. 89].

2.8 TraCI

Das Traffic Control Interface (TraCI) definiert den Zugriff auf eine laufende Straßenverkehrssimulation wie SUMO, anfordern von Simulationsobjekten und Manipulation der Simulationsobjekte zur Laufzeit[25]. TraCi definiert eine TCP basierte Client-Server

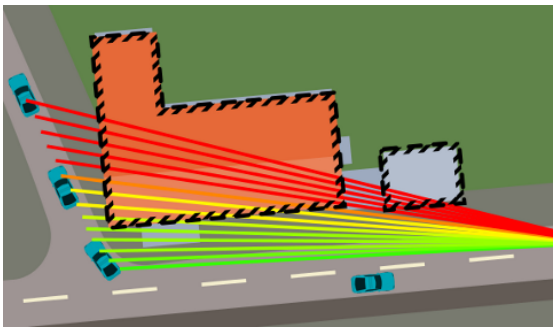


Abbildung 2.11: Schematische Signalabschwächung und Blockierung durch ein Gebäude[5, S. 84].

Architektur um Zugriff auf SUMO zu ermöglichen, dabei stellt SUMO den Server und Veins den Client dar. Um den Service zu benutzen muss SUMO mit dem Skript `python $HOME/veins-4.4/sumo-launchd.py -vv {-c sumo -gui}` gestartet werden. Das Skript `sumo-launchd.py` wird von dem Veins Framework zur Verfügung gestellt. Ein Überblick über den Aufbau von OMNeT++, Veins, SUMO und TraCI kann in der Abbildung 2.9 eingesehen werden, wobei TraCI der Pfeil ist, der Veins und SUMO verbindet.

2.9 INET

INET ist eine Open-Source Modell Library für Kommunikationsnetzwerke innerhalb der Simulationsumgebung OMNeT++. INET stellt Protokolle für Physical, Network, Transport, Application und Link Layer, sowohl kabelgebunden als auch drahtlose Protokolle wie IEEE 802.11, bereit[57].

2.10 Random Number Generator

Ein Random Number Generator (RNG) ist ein Generator für eine Sequenze von Zahlen, die nicht besser als mit einer zufälligen Wahrscheinlichkeit vorhergesagt werden kann[6]. Es wird in Random und Pseudo Random unterschieden, wobei die Zahlensequenzen aus dem Pseudo RNG sich eventuell wiederholen können, was bei einen RNG nicht möglich ist, da es z.B. von physikalischen Größen wie dem Wärmerauschen abhängig ist, welches wegen der Quanten Mechanik unvorhersehbar ist. In Computern werden

[6] https://en.wikipedia.org/wiki/Random_number_generation

meistens Pseudo Random Number Generator[7] (PRNG) verwendet um Variablen mit einen zufälligen Wert zu innitialisieren. Besonders in Simulationen ist dieses Vorgehen vorteilhaft, da durch die Wiederholungen eines Simulationsdurchlaufs die Variablen mit einen neuen Zufallswert initialisiert werden und eine unterschiedliche Auswirkung auf die Simulationsergebnisse haben. Daraus folgt das eine Simulation die Realität besser abbilden kann, wenn ein RNG Variablen zufällige Werte, in einem vorgegeben Intervall, zuweist und die Simulationsergebnisse über alle Wiederholungen gemittelt werden.

2.11 Lokalisierung

Ein Ortsbestimmungssystem dient dazu eine Aussage über den Aufenthaltsort eines Objektes innerhalb eines Koordinatensystem zu machen. Dazu kann in zwei Methoden unterschieden werden.

1. **Landmarken**

   Die Lokalisierung mit Landmarken verfolgt das Prinzip, dass eine Menge von Bezugspunkten im Koordinatensystem verteilt sind von denen die Koordinaten bekannt sind. Diese Bezugspunkte können sich auch innerhalb des Koordinatensystems bewegen und eine bestimmte Flugbahn verfolgen damit die Koordinaten zu jeden Zeitpunkt berechenbar sind. Um die Koordinaten eines Objektes zu bestimmen werden die Distanzen zu x Referenzobjekten bestimmt und es ergibt sich ein Gleichungssystem mit x Gleichungen und Unbekannten.

2. **Koppelnavigation**

   Die Koppelnavigation ist eine Akkumulation von bereits bekannten Koordinaten eines Objektes auf Basis von Bewegungsrichtung und Geschwindigkeit. Wenn ein Objekt sich von einen bekannten Punkt P aus in eine Richtung ϕ bewegt, mit einer Geschwindigkeit v, genau dann gilt für P'$(v * t * cos(\phi), v * t * sin(\phi))$. Durch die Akkumulation von Koordinaten findet eine Extrapolation mit ungenauen Daten statt, die in eine größer werdende Ungenauigkeit resultiert.

Ortsbestimmungssysteme zeichnen sich durch die Genauigkeit aus, die über den Aufenthaltsort eines Objektes gemacht wird, in Form von Koordinaten und durch die Geschwindigkeit mit der die Position aktualisiert wird. Folgend ist die Technik Lateration beschrieben die als Grundlage zur Positionsbestimmung das Prinzip der Landmarken verwendet.

[7] https://en.wikipedia.org/wiki/Pseudorandom_number_generator

2.11.1 Lateration

Durch die Entfernung zu einen oder mehreren Referenzobjekten wird über das Verfahren der Lateration die Koordinaten berechnet. Je mehr Referenzobjekte zum Zeitpunkt t verfügbar sind desto genauer kann die Position zum Zeitpunkt t bestimmt werden. Die Entfernung für die Positionsbestimmung wird durch das Verfahren Time of Flight oder Time Difference of Arrival bestimmt. Das Referenzobjekt sendet ein Datenpaket in dem die aktuelle Uhrzeit enthalten ist, nachdem die Nachricht empfangen wird kann eine Zeitdifferenz ermittelt werden. Die Zeitdifferenz ist die Zeit, die die Nachricht benötigt vom Sender zum Empfänger. Typischerweise wird die Geschwindigkeit als Lichtgeschwindigkeit c angegeben, sodass die Entfernung zum Referenzobjekt $d = c * \Delta t$ ist. Diese Technik hat den Nachteil, dass die Uhren aller Teilnehmer sehr exakt funktionieren müssen und Beeinträchtigungen des Signals nicht feststellbar sind. Mit der Methode Time Difference of Arrival wird das Problem mit ungenauen Uhren gelöst, indem zwei unterschiedliche Signale versendet werden und mit der Zeitdifferenz der Signale die Berechnung erfolgen kann. Mit der Formel $d = \Delta t(\frac{1}{v_1} - \frac{1}{v_2})$ wird die Entfernung bestimmt.

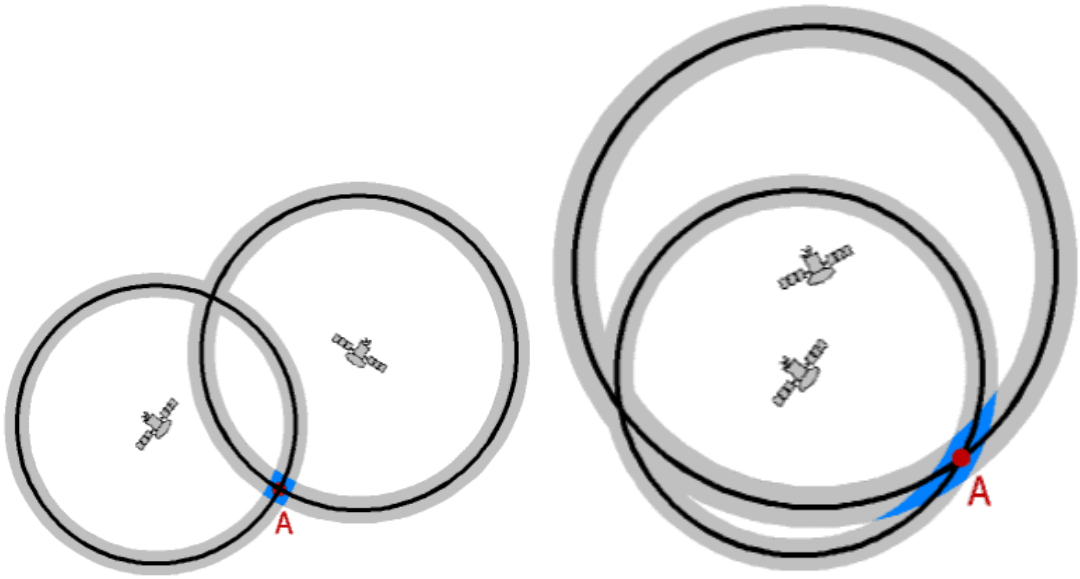
2.11.2 Mozilla Location Service

Mozilla Location Service (MLS)[11] ist ein Open Source Projekt, das weltweit die Signalstärken stationärer Sender in einer Momentaufnahme speichert. Die Koordinaten der Sender werden durch die Gemeinschaft approximiert, indem Momentaufnahmen aller verfügbaren Signale an die Mozilla Server gesendet werden. Durch Senden einer Momentaufnahme kann auf Basis der Signalstärke der jeweiligen Sender die eigene Position approximieren[11].

2.11.3 Global Positioning System

Global Positioning System (GPS) wird in dieser Arbeit als Oberbegriff für ein satellitengestütztes Navigationssystem zur Positionsbestimmung verwendet. Eine satellitengestützte Positionsbestimmung besteht aus mindestens 24 Satelliten, Kontrollstation auf der Erde und dem Benutzer, der den GPS Service in Anspruch nimmt. In den Satelliten ist eine Atomuhr, die sicherstellt, das alle Satelliten synchron und exakt funktionieren. Die Position wird durch die Übertragungszeit des Signals berechnet, sodass eine Genauigkeit von 10m eine Genauigkeit der Uhrzeit von $\frac{1}{30^9}s$ entspricht[12, S. 17]. Mit einer Ungenauigkeit von $\frac{1}{10^2}s$ wird die Position mit einer Ungenauigkeit von 3000Km aufgelöst[12, S. 17].

Durch die Positionsbestimmung mit mehreren Satelliten wird der Fehler minimiert. Für die Berechnung mittels Lateration müssen mehrere Satelliten verfügbar sein, die in einem idealen Winkel zueinander stehen, damit der Störradius klein ist(vgl. Abb.2.12).



(a) Zwei Satteliten mit einem idealen Winkel zu einander und kleinem Störradius.

(b) Zwei Satelliten mit einem ungünstigen Winkel und großem Störradius.

Abbildung 2.12: Satellitenpositionen zueinander[8].

Dabei erhöht sich die Genauigkeit mit jedem hinzukommenden Satelliten (vgl. Abb. 2.13). In der Ionosphäre und Troposphäre werden die Radiowellen teilweise reflektiert was in einer längeren Laufzeit resultiert[12, S. 21], was durch den Anwender nicht festgestellt wird ohne eine zweites Referenzsignal.

Die Kontrollstationen führen Kurskorrekturen durch, damit die Satelliten auf der vorgegebenen Umlaufbahn bleiben, sowie Uhrzeitanpassungen, denn durch die Zeitdilatation vergeht die Zeit für den Satelliten langsamer, als auf der Erde, sodass sich eine Ungenauigkeit von $\frac{480m}{h}$ für den Nutzer einstellt[14, S. 10].

[8] https://www.medien.ifi.lmu.de/lehre/ws0607/mmi1/essays/Andreas-Lodde/
[9] http://transform.gps2all.de/?cat=1

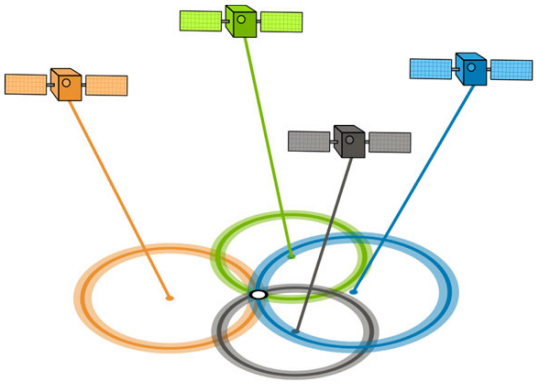


Abbildung 2.13: 4 Satelliten für eine genauere Positionsbestimmung[9].

3 Verwandte Arbeiten

Dieses Kapitel gibt einen Überblick über die Verwandten Arbeiten zu dieser Ausarbeitung und zeigt die jeweiligen Ergebnisse und Probleme auf, an denen sich diese Ausarbeitung orientiert.

Die Arbeit „An Advanced Vehicle Collision Warning Algorithm over the DSRC Communication Environment: An Advanced Vehicle Collision Warning Algorithm [64]" beschäftigt sich mit einen Algorithmus zur Kollisionswarnung, der die Probleme vorheriger Ansätze lösen soll. Zu den Probleme gehören die nicht konstante Geschwindigkeit der Verkehrsteilnehmer und kurvige Straßenverläufe die von bisherigen Systemen nicht gelöst sind. In der Veröffentlichung wird das Problem mit sich verändernden Geschwindigkeiten mit dem *Closest Point of Approach* (CPA) gelöst, indem in die zugehörigen Vektoren der Fahrzeuge die aktuelle Beschleunigung eingerechnet wird. Das zweite Problem unterteilt sich in zwei Teilprobleme bei dem ein Fahrer eine Kurve fährt, z.B. abbiegt, während der andere Fahrer unverändert weiterfährt und bei dem beide Fahrer in eine Kurve fahren. CPA ist die verwendete Methode bei dem der Lenkwinkel mit einbezogen wird. Beide Ansätze funktionieren wie bei dem ersten Problem beschrieben, mit dem Unterschied das der Lenkwinkel mit in den Vektor ein berechnet wird.

In dem Paper „Cooperative vehicle collision avoidance using inter-vehicle packet forwarding[65]" wird ein Szenario beschrieben, in dem durch Nachrichten eine Gefahr für eine Kettenkollision signalisiert wird und an alle betreffenden Fahrzeuge weiter gereicht wird. Da rote Bremslichter in der Regel nur von dem direkt folgenden Fahrzeugführer eingesehen wird und die Reaktion von Menschen deutlich langsamer ist als eine Nachricht zu versenden, so die Autoren, wird ein Kettenunfall verhindert, weil dem Fahrer mehr Zeit bleibt um auf die bevorstehende Situation angemessen zu reagieren. Dennoch kann in dem beschrieben Szenario die Situation nicht Unfallfrei aufgelöst werden, sondern der Schaden nur reduziert, weil der erste Fahrer in der Schlange keinen Vorteil durch die Gefahrennachricht hat.

In dem Paper „A Vehicle-to-Vehicle Communication Protocol for Cooperative Collision Warning[66]" wird ein System zur Unfallverhütung vorgestellt, beidem Fahrzeuge des

Netzwerkes Rollen zugeordnet sind. Verhält sich ein Fahrzeug unerwartet, z.B. durch starkes Bremsen, so geht es in den Zustand Äbnormal Vehicle"(AV) über und fängt an eine Gefahrenmitteilung zu versenden um folgende Fahrzeuge eine Gefahr zu signalisieren. Dies stellt besondere Anforderungen an die Chanel Kapazität, wenn mehrere AV unterschiedliche Gefahrenmitteilungen versenden.

In der Arbeit „A Deceleration Control Method of Automobile for Collision Avoidance based on Driver's Perceptual Risk[67]" wird diskutiert wie eine Situation erkannt wird um automatisch für den Fahrer zu bremsen. So soll automatisch eine präventive Bremsung eingeleitet werden, wenn z.B. der Abstand zum vorausfahrenden Fahrzeug zu gering ist.

In dem Paper „Design and Implementation of the Intelligent Stop and Go System in Smart Car, TAIWAN iTS-1[68]" wird beschrieben wie ein Fahrzeug, das mit Laser Scanner, Sensoren an dem Brems- und Gaspedal ausgestattet ist in drei beispielhaften Situationen Gefahren erkennt und selbstständig auflöst. Dazu gehören 4m vor Fußgängern anzuhalten, die die Straße überqueren, Hindernisse zu erkennen und 7m davor anzuhalten und eine Stop und Go Funktion die in Stauen verwendet wird um automatisch zum vorausfahrenden Automobil aufzuschließen.

4 Analyse

Dieses Kapitel befasst sich mit der Analyse der Aufgabenstellung dieser Arbeit. In Kapitel 4.1 wird festgelegt, wie das Ziel dieser Arbeit definiert ist. Anschließend ist im Kapitel 4.2 der Anwendungsbereich für das Assistenzsystem definiert, aus dem sich die Anforderung an das Szenario ableiten. Im Kapitel 4.3 wird eine Bestandsaufnahme für das Ziel dieser Arbeit durchgeführt, in der Diskutiert wird mit welcher Software, wie das Ziel dieser Arbeit erreicht wird. Außerdem wird diskutiert, ob und welche Einschränkungen durch die verwendete Software bestehen und welchen Einfluss die Einschränkungen auf die Simulationsergebnisse haben bzw. wie diese Einschränkungen umgangen werden. Im Kapitel 4.4 sind die Anforderungen an die Simulation, die Lokalisierung, den Protokoll Stack und das Assistenzsystem definiert. Abgeleitet von der Ist-Analyse und den Anforderungen sind im Kapitel 4.5 die Arbeitspakete festgelegt und erläutert, die zu bearbeiten sind um dem Ziel dieser Arbeit gerecht zu werden. Im Kapitel 4.6 wird die Methodik und Vorgehensweise festgelegt, mit der sichergestellt ist, dass die Simulationsergebnisse auf die Realität projizierbar sind. Es unterteilt sich in vier Unterkapitel. Das Abschnitt 4.6.1 beschreibt die Methodik, mit der die Auswahl der Iterationsvariablen festgelegt ist. Im Abschnitt 4.6.2 wird dargestellt nach welchen Kriterien und physikalischen Eigenschaften die simulierten Fahrzeuge ausgewählt sind. Die Auswahl und Beschreibung des simulierten Szenarios wird in Kapitel 4.6.4 aufgezeigt. Im Kapitel 4.6.3 wird das physikalische Verhalten unterschiedlicher Fahrzeugtypen erörtert und eine Auswahl von Fahrzeugtypen definiert. Im darauf folgenden Schritt wird die Zusammensetzung von vier Fahrzeugmengen aus den zuvor definierten Fahrzeugtypen diskutiert. Abschließend werden im Kapitel 4.7 die Hypothesen vorgestellt, die durch diese Ausarbeitung zu belegen sind.

4.1 Ziel dieser Arbeit

Ziel dieser Arbeit ist es, ein Assistenzsystem zu entwickeln, das in einem Szenario simuliert wird, in dem unterschiedliche Fahrzeugtypen Nachrichten austauschen. Basierend auf den empfangenen Nachrichten sind potenzielle Kollisionen zu erkennen und durch Bremsen

zu vermeiden. Die Simulation basiert darauf statistische Informationen zu generieren, die eine Aussage über die Nutzbarkeit der Anwendung unter unterschiedlichen Bedingungen macht. Die Qualität des Assistenzsystems soll danach bewertet werden, wie frühzeitig eine Kollision erkannt wird und wie viele Kollisionen entstehen, beides im Vergleich zu unterschiedlichen Simulationsdurchläufen, die durch unterschiedliche Parameterisierung unterschiedliche Bedingungen widerspiegeln. Es soll die Auswirkung der Parameter Reaktionszeit, Sendefrequenz, Mindestabstand und Positionsungenauigkeit auf die Leistungsfähigkeit untersucht werden. Weiterhin überqueren Ausnahmefahrer unbeachtet die Kreuzung und provozieren Kollisionen. Durch Integrieren von Ausnahmefahrern wird eine Übergangsphase simuliert. Ausnahmefahrer bilden die Realität realistischer ab, da nur eine Teilmenge aller Verkehrsteilnehmer ein Assistenzsystem nutzt und eine zweite Teilmenge nimmt passiv am Straßenverkehr teil durch broadcasten der Koordinaten, Fahrrichtung, Geschwindigkeit, Zeit, ID, Fahrzeuglänge und Beschleunigung. Weiterhin wird untersucht wie die Auswirkung des Assistenzsystems auf das Übertragungsmedium ist. Durch die Simulation sollen statistische Zusammenhänge der unterschiedlichen Parameterisierung erkannt werden, die sich auf die Kollisionen bzw. dem Erkennen der Killision auswirken um abschließend zu schlussfolgern, unter welchen Bedingungen ein beschriebenes Assistenzsystem für den Anwender nutztbar ist.

4.2 Anwendungsszenario

Die Arbeitsumgebung soll in großen Gemeinden bis Großstädten sein wie z.B. in Hamburg, Berlin oder Hong Kong. Dabei ist anzunehmen, dass die Durchschnittsgeschwindigkeit, je nach Uhrzeit, zwischen 10 und $50\frac{km}{h}$ ist. Es wird angenommen, dass dem Fahrzeugführer unter einer Sekunde Reaktionszeit verbleibt um mit einer Notbremsung oder einer anderen Aktion eine Kollision zu vermeiden. In Situation wie einer unübersichtlichen Kreuzung ohne Ampel, langen Kurven oder hügeligen Straßenverläufen kann die Reaktionszeit mehrere Sekunden betragen. Viel befahrene Streckenabschnitte ohne Ampeln, Abschnitte an Ampelanlagen und zugehörigen Kreuzungen, schwer einsehbare Abschnitte wie kurvige Verläufe, sowohl bei Tag und bei Nacht ohne Beleuchtung sind durch das zu entwickelnde System zu unterstützen.

4.3 Ist-Analyse

Für diese Bachelorarbeit ist ein Assistenzsystem zur Kollisionsverhütung entwickelt worden, dessen Anwendbarkeit im Straßenverkehr zu verifizieren ist. Dabei ist die Leistungsfähigkeit dieses Systems von unterschiedlichen Gegebenheiten abhängig, die sich durch die technischen Möglichkeiten der verwendeten Smartphones, physikalischen Eigenschaften der Automobile und Verkehrssituation ergeben. In einer Simulation wird die Realität in ein Modell überführt, auf dem zielgerichtet experimentiert wird, wodurch die Simulationsergebnisse zum Teil auf das reale Problem übertragbar sind (vgl. [43]). Die Vorteile einer Simulationsgestützten Verifikation gegenüber der realen liegt in der Sicherheit und der Kostenminimierung (vgl. [43]), die sich durch Versagen und Herstellung eines Prototyps ergeben können. Basierend auf dieser Argumentation wird im folgenden Kapitel eine Auswahl von Simulationssoftware getroffen und diskutiert.

4.3.1 Simulationssoftware

Das zu entwickelne Assistenzsystem basiert auf dem Austausch von Nachrichten, sodass es vorteilhaft ist, wenn die zugrundeliegende Simulationssoftware auch auf dem Austausch von Nachrichten basiert. Der Traffic, der durch das Assistenzsystem generiert wird, ist in zwei Bereiche zu unterscheiden; zum einen dem Netzwerk Traffic und zum anderen den Inter-Vehicular Traffic, sodass die verwendete Software die Simulation von beidem unterstützten muss. Weiterhin müssen die Automobile mobil sein und sich indeterministisch durch ein Straßenszenario bewegen, sodass die Simulationsumgebung die Realität nachbildet und dadurch die Forschungsergebnisse auf die Realität übertragbar sind. Die verwendeten Simulationswerkzeuge sollen eine zentrale Aufzeichnung von Daten für die statistische Auswertung ermöglichen.

OMNeT++

OMNeT++ (siehe 2.5) ist ein Simulationsframework, das auf dem Austausch von Nachrichten basiert, Modular erweiterbar ist und dadurch eine unbegrenzte Hierachie hat (vgl. 2.5.1). Durch die NED-Sprache können Funktionalitäten in Modulform miteinander kombiniert werden, dies erhöht die Wiederverwendbarkeit der Module und das Netzwerk wird mit minimalem Aufwand um Funktionalitäten, durch die Lose Kopplung der Module, erweitert (vgl. 2.5.2). Durch die Entkopplung der Generierung der Ergebnisse, von der Ergebniserfassung und weiterführender Manipulation im NED-File (vgl. 2.5.5 und 2.5.2), stellt OMNeT++ ein Instrument für die statistische Analyse bereit, die den oben

genannten Anforderungen genügt. Auf Grundlage dieser Argumentation wird OMNeT++ als Simulationsframework für die Netzwerksimulation verwendet. OMNeT++ soll in der Version 5.0 verwendet werden.

SUMO

SUMO (siehe 2.6) ist ein Verkehrssimulator, der das Simulieren jedes Fahrzeuges, durch Verhalten und zu fahrender Route, durchführt (vgl. 2.6). Durch individuelle Instantiierung und selbst entwickelte CFM[56] werden Verkehrsteilnehmer unter problemorientierten Einflüssen simuliert (vgl. 2.6 und 2.6.2). Durch das TraCI Interface (siehe 2.8) bietet SUMO die Möglichkeit auf die laufende Simulation Zugriff zu erhalten, um das Verhalten der Automobile zu manipulieren. Durch den Einfluß auf bestehende CFM, durch Anpassung von Parametern in der *vType* Beschreibung der Automobile (vgl. Tabelle 2.1), oder durch Entwicklung eines eigenem CFM, welches individuell für jedes Fahrzeug ausgewählt wird (vgl. 2.6.2), und den Zugriff auf die laufende Simulation über das TraCI Interface, genügt SUMO den oben gestellten Anforderungen an den Verkehrssimulator. SUMO soll in der Version 0.25.0 verwendet werden.

Veins

Veins ist ein Framework für die Simulationsumgebung OMNeT++, welches dieses um C2X Modelle erweitert (vgl. 2.7). Veins implementiert standardmäßig das IEEE 802.11p Protokoll, für die Modellierung von Physical- und MAC Layer (vgl. 2.7). Durch zusätzliche Protokolle wird der bestehende Protokoll Stack abgeändert und oder mit höheren Layern erweitert. Weiterhin modelliert Veins das Übertragungsmedium durch das *2RIM* und *OSM*, welches das Verhalten eines realen Übertragungsmediums vorhersagt, unter den Gesichtspunkten Signalüberlagerung und Signalabschirmung (vgl. 2.7.1). Veins implementiert das TraCI Interface (vgl. 2.8), wodurch zur Laufzeit Simulationsobjekte aus SUMO angefordert und in OMNeT++ instanziert werden, wodurch ein Datenaustausch der Automobile, auf Basis ihrer Eigenschaften, wie Position und Geschwindigkeit, in OMNeT++ simuliert wird (vgl. 2.7 und [4]). Durch die Instanziierung der Simulationsobjekte in OMNeT++, findet die Generierung von Daten -für die statistische Auswertung der Fahrzeuge- in OMNeT++ statt, sodass die Datenerfassung und Manipulation zentral im NED-File durchgeführt wird (vgl. 2.5.5). Veins verbindet die OMNeT++ Simulation mit der SUMO Simulation über das TraCI Interface, wodurch beide Frameworks bidirektional miteinander verbunden sind, das den Datenaustausch und die statistische Auswertung der Automobile zentral in OMNeT++ ermöglicht. Weiterhin stellt Veins die

Modulation von Übertragungsmedium und Layer 1 und 2, im OSI-Modell, bereit. Veins ist Modular durch höhere Protokolle erweiterbar, wodurch Veins den Anforderungen als Ausführungsumgebung für C2X-Anwendung genügt. Veins soll in der Version 4.4 verwendet werden, wodurch es mit OMNeT++ 5.0 und SUMO 0.25.0 kompatibel ist.

INET

INET ist eine Modell Library für OMNeT++ und erweitert diese um Protokolle für Kommunikationsnetzwerke (vgl. 2.9). Das INET Framework wird von der OMNeT++ Community weiterentwickelt und stellt Protokolle für die Layer 1,2,3,4 und 7 bereit (siehe 2.1, [57]). INET hilft dem Anwender für eine zu simulierende Anwendung den Protokoll Stack zu implementieren, indem Protokolle zur Verfügung stehen, welche in das eigene Projekt instantiiert werden. Mit der Version *Veins 4* sind INET und Veins kompatibel zueinander, sodass in einem Automobil ein INET Protokoll Stack instanziiert werden kann[58]. INET wird kontinuierlich durch die OMNeT++ Community weiterentwickelt, sodass es nicht als „vollständig" betrachtet wird und eine Anwendung von INET, zum Zeitpunkt der Ausarbeitung dieser Arbeit, zweifelhaft ist. Weiterhin ist INET nur bis zu der Versionsnummer 2.3 (März 2014) mit Veins kompatibel, welches das vorherige Argument der „Unvollständigkeit" verstärkt (vgl. [59], [60, Zeilen 16-40]).

4.3.2 Bewertung

In diesem Kapitel soll das Ergebnis der Ist-Analyse dargestellt und bewertet werden.

Car Following Modell

Das Krauss CFM ist, wie in Kapitel 2.6.1 bereits erörtert, das Standard CFM in SUMO und kollisionsfrei. Das Standardmodell muss abgeändert werden, oder ein neues Modell muss implementiert werden, welches den Anforderungen, die durch diese Arbeit gestellt sind, gerecht wird. Eine Abänderung bzw. eine Implementation von einem CFM wird durch SUMO garantiert[56], sodass diese Aufgabe im Kapitel 6 im Detail besprochen und bearbeitet wird.

Physikalisches Verhalten in SUMO

Die Straßenverkehrs Simulation, welche von SUMO durchgeführt wird, beinhaltet kein physikalisch korrektes Verhalten bei dem Eintreten einer Kollision. Wenn Automobile eine seitliche Kollision haben ($\approx 90°$ Winkel), dann fahren die Kollisionsteilnehmer

unverändert weiter. Wenn ein Auffahrunfall entsteht, zeigen beide Automobile keine Reaktion, sondern überlagern sich. Wenn die Automobile mehr als 2m überlagern, wird das hintere Fahrzeug aus der Simulation entfernt und in einen freien Straßenbereich, mit einer legalen Geschwindigkeit, auf der vordefinierten Route, in die Simulation eingefügt, wodurch sich die Lücke, zwischen zwei Automobilen, vergrößert bzw. verkleinert. Diese Eigenschaft von SUMO wird nicht ignoriert, dennoch liegt die Hauptfunktionalität des Assistenzsystems in dem frühzeitigen erkennen von potenziellen Kollisionen, sodass das physikalische Verhalten der Automobile, welches durch/nach eine(r) entstandende(n) Kollisionen hervorgerufen wird, eine geringfügige Auswirkung auf die Simulationsergebnisse hat, da diese Kollision bereits von den folgenden Automobilen registriert ist.

Veins

Veins stellt eine Schnittstelle zur Verfügung, womit das Anwenderprogramm einen kontextabhängigen Zugriff auf die Fahrzeugvariablen in SUMO hat. Für ein Fahren, aus der Sicht des Fahrers, welches durch das Assistenzsystem unterbrochen und manipuliert wird, muss der Zugriff auf alle Fahrzeugvariablen im Szenario möglich sein. Durch das Implementieren des TraCI Interfaces stellt Veins die Infrastruktur bereit, um mit SUMO zu kommunizieren und Daten auszutauschen. Mit der Infrastruktur soll das Veins Framework um Methoden erweitert werden, die den Zugriff auf die Automobilvariablen wie Geschwindigkeit, Fahrrichtung und Koordinaten ermöglichen. Diese Erweiterung ist ohne Einschränkung durchführbar, sodass diese Aufgabe im Kapitel 6 Implementierung im Detail besprochen und bearbeitet wird.

Statistische Auswertung

Durch die Möglichkeit Simulationsobjekte über eine TCP Verbindung von SUMO anzufordern und in OMNeT++ auszuwerten, ist die Analyse, wie in Kapitel 2.5.5 beschrieben, durchführbar.

Aufbau

Die simulierte Anwendung besteht aus zwei Teilbereichen die parallel ausgeführt werden; erstens das Assistenzsystem, das den Fahrer auf einen bevorstehende Kollision aufmerksam macht und zweitens die Fahrzeugführeranwendung, die auf Basis der exakten Daten der Simulationsobjekte sich durch das Szenario bewegt und auf Signalisierung durch das Assistenzsystem das Fahrverhalten anpasst. Die zwei Anwendungen sind durch

Selfmessages implementiert, wie im Kapitel 2.5.4 erläutert. Dieses Arbeitspaket wird im Kapitel 4.4 Anforderungen, 5 und 6 Implementierung im Detail besprochen und bearbeitet.

4.4 Anforderungen

Für diese Ausarbeitung sind in diesem Kapitel Anforderungen an die Lokalisierung, den Protokoll Stack, die Implementierung und die Simulation gestellt, abgeleitet von dem Ziel der Arbeit, dem Anwendungsszenario und der Ist-Analyse. Die Anforderungen werden gestellt, damit die Ausarbeitung zentralen und nachvollziehbaren Qualitätsstandards unterliegt. Folgend sind drei Anforderungen an die Lokalisierung, vier an den Protokoll Stack und an die Implementation des Assistenzsystems und fünf für die Simulation definiert.

Simulation

1. Die Simulation muss mit den Variablen Sendeintervall, Mindestabstand, Ausnahmefahrer, Reaktionszeit und Positionsungenauigkeit parametrisiert sein, damit die Auswirkung dieser elementaren Einflüsse auf das Assistenzsystem analysierbar sind. Für diese Variablen müssen Werte verwendet werden die dem *Ist-Zustand* und einem *Zukunftswert* entsprechen, damit die Leistungsfähigkeit und Anwendbarkeit des Assistenzsystems einzuordnen ist.

2. Es muss jede Kombination der Variablen simuliert werden und jede Kombination muss ausreichend oft wiederholt werden, damit die Ergebnisse statistisch relevant sind.

3. Das physikalische Verhalten der Automobile in der Simulation muss dem eines realen Fahrzeuges entsprechen für die maximale Beschleunigung und Größe.

4. Es muss sichergestellt werden, dass die Simulationen indeterministisch sind.

5. Das verwendete Szenario muss den Eigenschaften eines Stadtszenarios entsprechen. Der Verkehr muss ausreichend dicht sein, die durchschnittliche Geschwindigkeit muss dem in einer Stadt entsprechen, das Szenario muss Kurven und eine Kreuzung enthalten und die Signale im Übertragungsmedium müssen durch die Automobile, Häuser und dem Boden beeinflusst werden.

Lokalisierung

6. Für die Lokalisierung muss eine Aktualisierungsrate ausgearbeitet und eingehalten werden.
7. Für die Lokalisierung muss eine größte Ungenauigkeit der Position ausgearbeitet und eingehalten werden.
8. Die Lokalisierung muss Infrastruktur unabhängig sein.

Protokoll Stack

9. Der Physical Layer muss die verfügbare Bandbreite effizient nutzen.
10. Der MAC Algorithmus muss eine Worst-Case-Kanalzugriffszeit bereitstellen.
11. Der MAC Algorithmus muss einen effizienten Kanalzugriff bereitstellen.
12. Der Protokoll Stack muss für die Anforderung 5 an das Szenario optimiert sein.

Assistenzsystem

13. Durch die Analyse der empfangenen Nachrichten, soll die Früherkennung von Kollisionen durch eine größer werdende Positionsungenauigkeit mindestens konstant bleiben, bis zu einer definierten Obergrenze.
14. Wenn eine Kollision erkannt wird, sollen die Verkehrsteilnehmer so stark wie nötig und wenig wie möglich bremsen, damit die Distanz zur Kollision für alle folgende Verkehrsteilnehmer effizient ausgenutzt wird.
15. Das Assistenzsystem muss Kollisionen erkennen auf Basis von ausgetauschten Nachrichten der Verkehrsteilnehmer.
16. Das Assistenzsystem muss Kollisionen erkennen, die für den Fahrer (noch) nicht zu erkennen sind, sodass der Fahrer früher auf eine bevorstehende Kollision reagiert.

Diese Ausarbeitung orientiert sich an diesen Anforderungen, damit die Qualität dieser Arbeit gesichert ist.

4.5 Arbeitspakete

Die zugrunde liegende Arbeit ist in fünf Arbeitspakete unterteilt, die sich aus der Ist-Analyse 4.3 und den Anforderungen 4.4 ergeben. Die vorgestellten Arbeitspakete werden im Kapitel Konzept 5 und Implementierung 6 im Detail vorgestellt und bearbeitet.

1. Erweiterung des Veins Framework um Methoden die Details über die Simulationsobjekte anfordern.
    a) Durch Veins wird bis zur Version 4.4 nur im Kontext eines konkreten Fahrzeugs Informationen durch SUMO angefordert. Dies soll abgeändert werden, sodass Kontextunabhängig jedes Fahrzeug Informationen über alle Fahrzeuge in der Simulation anfordern kann. Dieses Arbeitspaket ist die Grundlage für das Car Following Modell im folgendem Arbeitspaket.
2. Das Car Following Model (CFM) im SUMO Framework anpassen.
    a) Das Krauss CFM, das standardmäßig in SUMO angewendet wird[26], ist ein Kollisionsfreies Modell[27]. Im Rahmen dieser Arbeit wird ein CFM benötigt, das Fahren aus der Perspektive des Fahrers simuliert und Kollisionen toleriert. Dazu muss das angepasste Modell Geschwindigkeitsänderungen aus dem Anwenderprogramm entgegen nehmen, validieren und anwenden.
3. Den Fahrzeugführer entwickeln.
    a) Das Fahren aus der Sicht des Fahrzeugführers wird durch Analyse der exakten Koordinaten aller Netzwerkteilnehmer erreicht und baut auf dem Arbeitspaket 1 auf. Ist der festgelegte Abstand zum vorausfahrenden Automobil, das nicht abbremst, ungleich dem aktuellen Abstand, wird das Fahrzeug beschleunigt oder gebremst.
4. Das Assistenzsystem entwickeln.
    a) Es muss periodisch ein Update Broadcast generiert und versendet werden und die empfangenen Broadcasts müssen in einer Datenstruktur sortiert nach dem Sender abgelegt werden.
    b) Durch die Analyse der empfangenen Nachrichten soll die Kollisionswahrscheinlichkeit mit steigender Positionsungenauigkeit nicht zu nehmen. Dieses Arbeitspaket implementiert die Anforderung 13 im Kapitel 4.4.

c) Basierend auf der Anforderung 14 im Kapitel 4.4 soll ein Algorithmus entwickelt werden, der das Automobil so stark wie nötig und gering wie möglich abbremst nachdem erkennen einer Kollision.

d) Basierend auf den Empfangenen Nachrichten werden Kollisionen erkannt und dem Fahrer signalisiert, worauf dieser nach einer Reaktionszeit reagiert. Dieses Arbeitspaket implementiert die Anforderung 15 im Kapitel 4.4

5. Für die statistische Auswertungen müssen Ereignisse protokolliert werden.

a) Es soll protokolliert werden, wann eine bevorstehende Kollision dem Fahrer signalisiert wird und wenn eine Kollision entstanden ist. Die Anzahl der entstandenen Kollisionen und die Zeit bis zu der bevorstehenden Kollision sind direkte Indikatoren der Leistungsfähigkeit des Assistenzsystems unter unterschiedlichen Bedingungen.

b) Die Anzahl der korrekt und fehlerhaft empfangen Broadcasts. An den korrekt und fehlerhaft empfangenen Broadcasts wird erkannt wie ausgelastet das Übertragungsmedium unter unterschiedlichen Bedingungen ist.

c) Die Anzahl der gesendeten Broadcasts Nachrichten soll dokumentiert werden, weil es aufzeigt wie ausgelastet das Übertragungsmedium ist.

d) Es soll Protokolliert werden, wenn der Sendethread im Backoff Algorithmus ist (siehe 2.2.2).

e) Die Anzahl der gewarteten Slots auf dem Übertragungsmedium soll dokumentiert werden, weil es aufzeigt wie ausgelastet das Übertragungsmedium ist.

f) Die Zeit und gefahrene Strecke im Szenario soll dokumentiert werden, damit das Verhalten der Fahrzeuge in der Simulation analysiert werden kann.

4.6 Methodik

In diesem Kapitel soll die Methodik erläutert werden mit der diese Arbeit angefertigt ist. Durch die vorgestellte Methodik ist sichergestellt, dass die Simulationsergebnisse auf die Realität projizierbar sind und somit die Aussagekraft haben, die im Kapitel 4.7 genannten, Hypothesen zu belegen.

In der Tabelle 4.1 sind die Parameter einzusehen, mit der das Assistenzsystem parametersiert ist. Durch Parametrisierung ist das Assistenzsystems unter unterschiedlichen

Bedingungen simuliert, wodurch analysierbar ist, welche Auswirkung jeder Parameter auf die Leistungsfähigkeit hat.

Variable	**Beschreibung**	**Wert**
Sendeintervall	Gibt an wie oft pro Sekunde (Hz) ein Update Broadcast gesendet wird.	10, 2, 1
Reaktionszeit	Gibt die Reaktionszeit des Fahrzeugführers in Sekunden (s) an.	0.6, 1
Mindestabstand	Gibt den minimalen Abstand zum vorausfahrenden Fahrer in Sekunden (s) an.	1.0, 2.0
Positionsungenauigkeit	Gibt die maximale Ungenauigkeit der aktuellen Position als Radius in Meter (m) an.	0.1, 2.0
Ausnahmefahrer	Prozentualer (%) Anteil aller Verkehrsteilnehmer die das Assistenzsystem nicht nutzt, aber ihre Position broadcastet. Ausnahmefahrer überqueren unbeachtet die Kreuzung.	20, 10, 5
Wiederholungen	Gibt die Anzahl der Wiederholungen eines Simulationsdurchlaufs mit den selben Parametern an.	20

Tabelle 4.1: Iterationsvariablen für die Simulationsdurchläufe.

4.6.1 Diskussion der Iterationsvariablen

In diesem Abschnitt wird die Parameterauswahl der Tabelle 4.1 und die zugehörigen Werte diskutiert. Die Parameter *Sendeintervall*, *Reaktionszeit*, *Abstand*, *Positionsungenauigkeit* und *Ausnahmefahrer* haben unterschiedliche Auswirkungen auf die Anwendbarkeit des Assistenzsystems und, daraus resultierend, auf die Sicherheit im Straßenverkehr. Die Parameter übernehmen die Funktion wie in Tabelle 4.1 beschrieben. Im Folgenden wird die Werteauswahl der jeweiligen Parameter erörtert und begründet.

1. Sendeintervall

- Der Broadcast soll in einer Frequenz von 10Hz, 2Hz und 1Hz gesendet werden. Das Sendeinterval richtet sich nach der Frequenz, mit der die Position aktualisiert wird. 1Hz entspricht dem Standardintervall des GPS Empfänger in jedem Smartphone[29, 31, S. 1ff]. Smartphones wie das Samsung Galaxy S5 haben den NAVSTAR, GLONASS und Beidou GPS Empfänger verbaut[30] wodurch ein Intervall von 2Hz erreicht wird. 10Hz entspricht einem Zukunftswert für ein Assistenzsystem, das auf dem GPS Signal basiert mit einer hohen Auflösung der Position. Bei einer maximalen Geschwindigkeit von $14\frac{m}{s}$ ist die Auflösung bei 10Hz im einstelligen Meterbereich und bei $7\frac{m}{s}$ im zweistelligen Zentimeterbereich. Basierend auf dieser Argumentation wird die Anforderung 6 im Abschnitt 4.4 eingehalten.

2. Reaktionszeit
   - Die Reaktionszeit beträgt eine Sekunde[32, S. 3] und 0,6 Sekunden[32, S. 3]. Eine Reaktionszeit von einer Sekunde entspricht der eines Fahrers, dessen Aufmerksamkeit nach der Signalisierung durch das Assistenzsystem fokusiert ist und auf visueller Wahrnehmung gebremst wird. Eine Reaktionszeit von 0,6 Sekunden entspricht dem sofortigen bremsen auf signalisierung durch das Assistenzsystem.

3. Mindestabstand
   - Der Mindestabstand von 2s wird in Fahrschulen gelehrt und ist ein Indikator für einen sicheren Abstand zum vorausfahrenden Verkehrsteilnehmer. Der Mindestabstand von einer Sekunde soll den minimalen Extremfall darstellen.

4. Positionsungenauigkeit
   - In dem „Performance Analysis Report[40]" aus dem Jahr 2014 ist die Leistungsfähigkeit des NAVSTAR GPS statistisch ausgearbeitet. Zu 95% wird eine höhere Genauigkeit als 4,684m Vertical und 3,351m Horizontal erreicht[40, S. 22]. Weiterhin ist die maximale Ungenauigkeit der berechneten Position bekannt[33], sodass die Ungenauigkeit der Position in dem Intervall, das im Performace Analysis Report[40, S. 22] erörtert ist, keine signifikante Auswirkung auf das Assistenzsystem hat. Aus diesen genannten Grund wird die Positionsungenauigkeit mit den Werten von 2m und 0,1m in die Simulation integriert. 2m entsprechen der regulären Auflösung, die durch ein Kombiniertes GPS System wie Galileo, GLONASS und NAVSTAR erreicht wird[34].

NAVSTAR GPS stellt für US-militärische Anwendungen eine höhere Auflösung bereit, die für die Öffentlichkeit nicht zugänglich ist[41]. Durch Broadcasten in zwei unterschiedlichen Frequenzbereichen wird die Signalschwächung durch die Ionospähre vermindert, wodurch eine höhere Auflösung möglich ist[41]. Da eine technische Realisierung einer höheren GPS Genauigkeit möglich ist wird der Wert von 0,1m in die Simulation als Zukunftswert integriert. Basierend auf dieser Argumentation wird die Anforderung 7 im Abschnitt 4.4 eingehalten.

5. Ausnahmefahrer
   - Ausnahmefahrer überqueren Kreuzungen unbeachtet und provozieren Kollisionen. Der Anteil von Ausnahmefahrern ist Lokal unterschiedlich, sodass für die Simulation die Werte 20%, 10% und 5% ausgewählt sind.
6. Wiederholungen
   - Deterministische Simulationen ergeben mit jedem Durchlauf die gleichen Ergebnisse, sodass keine Wiederholungen nötig sind[35, S. 1]. Diese Simulation ist indeterministisch, weil Parameter von Zufallswerten aus einem Zufallsgenerator abhängig sind, die mit jeder Wiederholung einen unterschiedlichen Wert in einen vorgegebenen Intervall möglicher Werte einnehmen. Dieses Vorgehen resultiert in unterschiedliche Auswirkungen auf die Simulationsergebnisse, sodass ein Mittelwert aller Wiederholungen eine bessere Abbildung der Realität entsprechen. Für die Simulation ist das Verhalten von 74 Fahrzeugen (vgl. Tabelle 4.3) simuliert, dass am Ende jeder Simulation für jede Statistik (vgl. Kapitel 4.5, Arbeitspaket 4) eine Stichprobe darstellt. 72 Simulationen mit 20 Wiederholungen generieren 106560 Stichproben für jede Statistik (vgl. Kapitel 4.5, Arbeitspaket 4), die für die Auswertung dieser Arbeit als ausreichend bewertet werden.

Durch das Parametrisieren der Simulation mit den Iterationsvariablen Sendeintervall, Mindestabstand, Reaktionszeit, Ausnahmefahrer und Positionsungenauigkeit und die zugehörige Diskussion und Auswahl der konkreten Werte, die einen Ist-Zustand und Zukunftswert oder optimierten Zustand repräsentieren wird die Anforderung 1 im Abschnitt 4.4 eingehalten. Durch die Auswahl von 20 Wiederholungen jedes Simulationsdurchlaufs wird weiterhin die Anforderung 2 eingehalten.

4.6.2 Auswahl der Verkehrsteilnehmer

Die Leistungsfähigkeit des Assistenzsystems soll unabhängig von der verwendeten Fahrzeugklasse sein, sodass für die Simulation eine Varietät an Eigenschaften der Fahrzeuge gewählt wird.

Type	**Beschleunigung in $\frac{m}{s^2}$**	**Geschwindigkeit in $\frac{m}{s}$**	**Länge in m**
1	± 8	14	6
2	± 6	12	8
3	± 10	16	3

Tabelle 4.2: Auflistung der unterschiedlichen Fahrzeugklassen und ihrer Eigenschaften

In Tabelle 4.2 sind die Fahrzeugtypen und ihre Eigenschaften einzusehen. Die Fahrzeugtypen repräsentieren kein Fahrzeugmodell, sondern sollen eine Varietät von Eigenschaften unterschiedlicher Fahrzeugmodelle aufweisen. Type 1 ist ein 6m langes Fahrzeug mit einer maximalen Beschleunigung von $\pm\frac{8m}{s^2}$ und einer maximalen Geschwindigkeit von $\frac{14m}{s}$. Type 1 repräsentiert die Fahrzeugklasse der Familienautos, Kombinationskraftwagen und Limousinen (vgl. [36, 37, S. 4]). Type 2 ist ein 8m langes Automobil, mit einer maximalen Geschwindigkeit von $\frac{12m}{s}$ und maximaler Beschleunigung von $\pm\frac{6m}{s^2}$. Type 2 repräsentiert die Fahrzeugklasse der Transporter, Busse und Kleinbusse (vgl.[37, S. 4]). Type 3 ist ein 3m langes Fahrzeug, mit einer maximalen Geschwindigkeit von $\frac{10m}{s}$ und maximalen Beschleunigung von $\pm\frac{10m}{s^2}$. Type 3 repräsentiert die Klasse der Motorräder, Kleinkrafträder und kleinen Automobilen (vgl.[37, S. 4]). Zusammenfassend repräsentiert der Type 1 die Fahrzeugklasse, die am häufigsten auf deutschen Straßen fährt. Type 2 repräsentiert ein großes Fahrzeug, welches den Verkehrsfluss beeinträchtigt und Type 3 die Klasse der kleinen und schnellen Verkehrsteilnehmer. Die maximale Geschwindigkeit der jeweiligen Fahrzeugen in der Simulation wird durch die Variablen *speedFactor* und *speedDev* in der vType Beschreibung in SUMO bestimmt[42]. Mit dem Erwartungswert als *speedFactor* und der Standardabweichung als *speedDev* wird eine maximale Geschwindigkeit implementiert, bei der 95% der Fahrzeuge zwischen 80% und 120% der erlaubten Geschwindigkeit fahren, indem die maximale Geschwindigkeit der Straße, auf der sich das Fahrzeug befindet, mit dem Wert aus der Verteilung multipliziert wird[42]. Durch die Verteilung der maximalen Geschwindigkeit ist das Verhalten der jeweiligen Fahrzeuge indeterministischer.

Durch die Auswahl der Parameter *Beschleunigung* und Größe der Fahrzeuge, welche sich an den physikalischen Eigenschaften von realen Fahrzeugen wie Kombinationskraftwagen,

Limousinen, Transporter, Bussen und Motorrädern orientiert wird die Anforderung 3 im Kapitel 4.4 eingehalten. Weiterhin wird durch die Verteilung der maximalen Geschwindigkeit das Verhalten der Automobile indeterministisch, sodass die Anforderung 4 eingehalten wird.

4.6.3 Fahrzeugmengen

Im Kapitel 4.6.2 sind vier unterschiedliche Fahrzeugtypen beschrieben. In der Tabelle 4.3 sind vier Fahrzeugmengen definiert, die eine unterschiedliche Zusammensetzung der Fahrzeugtypen aus Tabelle 4.2 zeigt.

Fahrzeugmenge	**Type 1 (in %)**	**Type 2 (in %)**	**Type 3 (in %)**	**Gesamt (in %)**
1	10 (62,5)	2 (12,5)	4 (25,0)	16 (21,6)
2	10 (50,0)	2 (10,0)	8 (40,0)	20 (27,0)
3	10 (45,0)	4 (18,0)	8 (36,0)	22 (29,7)
4	10 (62,5)	2 (12,5)	4 (25,0)	16 (21,6)

Tabelle 4.3: Zusammensetzung der Fahrzeugmengen aus unterschiedlichen Fahrzeugtypen

Die Tabelle 5.1 zeigt die Zusammensetzung der Fahrzeugmengen aus unterschiedlichen Fahrzeugtypen. Die Zusammensetzung der Fahrzeugmengen ist inhomogen gewählt, sodass ein realitätsnahes Abbild erreicht wird. Der prozentuale Anteil eines Fahrzeugtypes ist mit einer Tendenz gewählt, sodass der Type 1 mit 54% der primäre Fahrzeugtype ist, Type 2 mit 13,5% den kleinsten Anteil der Fahrzeuge im Szenario stellt und Type 3 mit 32,4% in der Mitte von Type 1 und 2 liegt. In Deutschland sind 49 Millionen Pkw, Lkw, Busse und Motorräder zugelassen, wobei 83% davon Autos sind[38] (vgl. Type 1 & 2). Für diese Ausarbeitung ist der Anteil der Typen 1 und 2 auf 67,5% limitiert, da in süd-ost Asien der Anteil der Motorräder und Kleinkrafträder im Vergleich zu Deutschland größer ist, sodass dieses Szenario übertragbar ist auf unterschiedliche Verkehrszusammensetzungen.

Jeder Fahrzeugtyp, jeder Fahrzeugmenge wird in definierten Zeitintervallen in die Simulation eingefügt, die in Tabelle 4.4 einzusehen sind. Die Menge 3 und 4 haben einen späteren Startzeitpunkt als die Mengen 1 und 2, da die Fahrzeuge der Menge 1 und 2 eine größere Strecke zu der Kreuzung zurücklegen und somit sichergestellt ist, dass die ersten Fahrzeuge gleichzeitig die Kreuzung überqueren. Alle Fahrzeugtypen einer Menge

werden sequenziell mit einer Zeitdifferenz von 2s und im Zeitintervall in die Simulation eingefügt. Zwei Sekunden ist der Mindestabstand, den die Fahrzeuge benötigen, damit keine Kollision durch das Einfügen entsteht. Dieses Vorgehen erzeugt einen inhomogenen und gruppierten Verkehr. Alle Fahrzeugtypen der jeweiligen Fahrzeugmengen haben das gleiche Zeitintervall, sodass in jeden Simulationsdurchlauf auch eine unterschiedliche Dichte an Verkehrsaufkommen simuliert wird. In jedem Simulationsdurchlauf steigt der Verkehr an und erreicht einen Höchststand, anschließend wird Type 2 und 3 nicht mehr in die Simulation eingefügt und die Verkehrsdichte nimmt ab. Wenn alle Fahrzeuge des Types 1 aus der Simulation entfernt sind ist der Simulationsdurchlauf beendet. Das erste Fahrzeug wird zum Zeitpunkt $t_0 = 0$ in die Simulation eingefügt und das letzte zum Zeitpunkt $t_1 = 143$. Zum Zeitpunkt t_1 wird das Szenario mit einer durchschnittlichen Verkehrsdichte von 74 Verkehrsteilnehmern in 143s simuliert und entspricht $\frac{1863\ Fahrzeugen}{h}$. Als Vergleich soll der Streckenabschnitt der Bundesstraße 6 (Zählstellennummer 3321 0411) bis Bundesstraße 6/214 in Nienburg (Weser) aus dem Jahr 2010 betrachtet werden. Dort ergibt sich eine Verkehrsdichte von 17.700 Fahrzeugen pro Tag, welches 738 Fahrzeugen pro Stunde entspricht (vgl. [39]).

Fahrzeugmenge und Type	**Startzeit (Sekunden)**	**Intervall (Sekunden)**
Menge 1, Type 1	2	14
Menge 1, Type 2	4	14
Menge 1, Type 3	6	14
Menge 2, Type 1	0	14
Menge 2, Type 2	4	14
Menge 2, Type 3	2	14
Menge 3, Type 1	17	14
Menge 3, Type 2	21	14
Menge 3, Type 3	19	14
Menge 4, Type 1	16	14
Menge 4, Type 2	20	14
Menge 4, Type 3	18	14

Tabelle 4.4: Zeitpunkt, zu dem die jeweiligen Fahrzeugtypen in die Simulation eingefügt werden in Abhängigkeit zum zugehörigen Zeitintervall.

4.6.4 Auswahl des Szenarios

Das Szenario soll wie im Kapitel 4.2 gestaltet sein. Das Szenario ist in Abbildung 4.1 zu sehen. Das Szenario enthält kurvige Streckenabschnitte und eine Kreuzung. An den Straßen stehen Häuser (in rot eingefärbt), die die ausgesendeten Signale reflektieren und abschwächen, wodurch physikalisches Verhalten simuliert ist. Die 1. Menge an Fahrzeugen wird im süd-osten in die Simulation eingefügt, fährt um die Kurve, über die Kreuzung und werden östlich der T-Kreuzung aus der Simulation entfernt. Die 2. Menge an Fahrzeugen startet im Norden, fährt über die Kreuzung und wird im Süden aus der Simulation entfernt. Die 3. und 4. Menge fahren entgegengesetzt zu der Menge 1 bzw. 2. Die maximale Geschwindigkeit der Straßen im Szenario ist mit $\frac{14m}{s}$ festgelegt.

Die zu fahrende Strecke für die Mengen 1-4 sind 400m, 450m, 350m und 350m, sodass eine gewichtete arithmetisches Distanz von 390,54m zu fahren ist (vgl. Tabelle 4.3). Aus der Tabelle 4.4 ergibt sich eine Zeit von 126 Sekunden zwischen dem zuerst und zuletzt eingefügten Automobil der vier Fahrzeugmengen. Weiterhin ergibt sich aus der Tabelle 4.2 eine gewichtete arithmetische Fahrzeuglänge von 5,5m, 5,27m, 5m und 5,5m für die Fahrzeugmengen 1,2,3 bzw. 4, sodass die Fahrzeugdichte mit der Formel 4.1 berechnet wird, wobei i die Laufvariable für die Fahrzeugmenge ist, die zugehörige Anzahl an Automobilen und die gewichtige arithmetische Fahrzeuglänge. Die Formel 4.1 besitzt die Einheit $\frac{\mathbf{V}ehicle}{\mathbf{S}ekunde}$.

$$Vehicle\ Density\left[\frac{V}{s}\right] = \frac{\sum_{i=1}^{4} \frac{Vehicles_i * VehicleLength_i}{\Delta Time_i}}{4} \tag{4.1}$$

Aus der Formel 4.1 folgt für das Szenario eine Automobildichte von $0,815\frac{V}{s}$ bzw. $1,23\frac{s}{V}$, also die zeitliche Distanz von z.B. vorderer Stoßstange zu vorderer Stoßstange des vorausfahrenden Automobils.

Der Abschnitt 4.6.3 hat gezeigt wie die Fahrzeugmengen zusammengesetzt sind und wie diese in das Szenario eingefügt werden. Das Szenario weist eine maximale Geschwindigkeit von $\frac{14m}{s}$ auf, eine Kreuzung und Kurven und eine Fahrzeugdichte von 1863 Automobilen pro Stunde. Außerdem stehen Gebäude in dem Szenario, welche die Signale blockieren und abschwächen, sodass die Anforderung 5 im Kapitel 4.4 eingehalten ist.

Abbildung 4.1: Das simulierte Szenario für diese Ausarbeitung.

4.7 Hypothesen

In dem Kapitel 4.6 ist die Methodik ausgearbeitet, mit der diese Arbeit durchgeführt ist. Durch diese Vorgehensweise sind die Simulationsergebnisse auf die Realität übertragbar, sodass mit dieser Ausarbeitung die folgenden Hypothesen zu belegen sind.

1. Wenn dem Assistenzsystem mehr Broadcast Status Updates zur Verfügung stehen, dann sinkt die Anzahl der Kollisionen, weil Kollisionen früher erkannt werden.
2. Wenn die Positionsungenauigkeit[1] größer wird, dann sinkt die Anzahl der Kollisionen, weil Kollisionen früher erkannt werden.
3. Wenn der Mindestabstand größer wird, dann sinkt die Anzahl der Kollisionen, weil Kollisionen früher erkannt werden.
4. Wenn die Reaktionszeit des Fahrers sinkt, dann sinkt die Anzahl der Kollisionen.
5. Wenn die Anzahl der Assistenzsystem-Nutzer steigt, dann sinkt die Anzahl der Kollisionen.

Auf Basis der belegten Hypothesen im Kapitel 7 und der aktuellen Technik findet eine Einordnung dieser Technologie unter den Kriterien der Anwendbarkeit und Leistungsfähigkeit im Kapitel 8 statt.

[1] In der Größenordnung, wie im Abschnitt 4.6.1 ausgearbeitet, wegen der Anforderung 9 für die Positionsungenauigkeit im Kapitel 4.4.

5 Konzept

In diesem Kapitel wird das Konzept besprochen, mit dem die Implementierung des Assistenzsystem durchgeführt wird. Das Konzept orientiert sich an den Anforderungen die durch den Anwendungsbereich und das Szenario gegeben sind. Der Anwendungsbereich stellt die Anforderung von einer Geschwindigkeit von $\frac{50m}{s}$, bei der durch den Doppler-Effekt[1] der Frequenzbereich der ausgesendete Signale verschoben wird, Hauswände und Automobile die Signale reflektieren und ein Übersprechen herbeiführen. Außerdem ist ein Stadtszenario gewählt mit Kurven und einer Kreuzung, sodass die Automobile im Szenario Anforderungen an die Kanaleffizienz haben und eine Worst-Case-Kanalzugriffszeit von 10ms[48] eingehalten sein muss. Das Konzept teilt sich in die Abschnitte *Lokalisierung* 5.1 auf, welches sich mit der Erlangung der Koordinaten in Abhängigkeit von der Anforderungen im Kapitel 4.4 beschäftigt. Dem *Protokoll Stack* 5.2, indem eine Auswahl von Protokollen zu treffen ist, die konform mit den Anforderungen an das Szenario und den *Protokoll Stack* im Kapitel 4.4 sind. Das Assistenzsystem 5.3, welches das Konzept unter Berücksichtigung der Anforderungen in 4.4 für die Teilbereiche der Assistenzsystemanwendung ausarbeitet. Das Kapitel *Statistik* befasst sich mit den zu erfassenden Daten in der Simulation und bespricht wie die Hypothesen aus 4.7 bewiesen werden.

5.1 Lokalisierung

Für das Assistenzsystem werden die Koordinaten von jedem Teilnehmer benötigt, sodass sich dieses Kapitel mit den Techniken beschäftigt, wie die Verkehrsteilnehmer ihre Koordinaten erlangen. Es wird die Einsatzfähigkeit von MLS diskutiert, welches auf dem Prinzip der Landmarken basiert, indem die eigene Position durch eine Menge von bekannten Sendestationen approximiert wird (vgl. 2.11.2). Anschließend findet eine Diskussion über das GPS statt, welches durch eine Menge von Satelliten in der

[1] Zeitliche Dehnung bzw. Stauchung eines Signals, wenn der Abstand zwischen Sender und Empfänger, während der Signallaufzeit, verändert wird. https://de.wikipedia.org/wiki/Doppler-Effekt

Erdumlaufbahn mit den Techniken Lateration und Time of Flight (siehe 2.11.1) die Koordinaten approximiert.

5.1.1 Mozilla Location Service

Mozilla Location Service (MLS, siehe 2.11.2) ist ein Open Source Projekt, welches im Jahr 2016 bereits auf 601 Millionen Datensätze Zugriff bietet (vgl. [11]). Die Vorteil von MLS ist, dass die Position in Städten sehr gut bestimmt wird, da in dicht besiedelt Städten mehr Sender wie WLAN, Bluetooth und andere stationäre Einheiten verfügbar sind. Der Nachteil von MLS ist, dass außerhalb der Stadt die Verfügbarkeit von stationären Sendeeinheiten sich verringert, sodass die Qualität der Lokalisierung abnimmt und die Anforderung 8 im Abschnitt 4.4 nicht eingehalten wird. Außerdem findet eine Anfrage über das Internet statt, welche das Laufzeitverhalten beeinflusst und eine Infrastruktur voraussetzt, wodurch die Anforderung 6 nicht garantiert und 8 nicht eingehalten wird im Abschnitt 4.4.

5.1.2 GPS

Global Positioning System ist ein Lokalisierungsservice, welches auf Satelliten in der Erdumlaufbahn basiert (GPS, siehe 2.11.3). Ein GPS System kann, am Beispiel des US NAVSTAR, eine Auflösung der Koordinaten erreichen, welche zu 95% eine höhere Genauigkeit als 4,684m in der Vertikalen und 3,351m in der Horizontalen bereitstellt.[13, S. 22]. Die Genauigkeit der Koordinaten und wie oft in der Sekunde die Position erneuert wird sind Leistungsindizes für die Qualität des Lokalisierungsservice. Standardmäßig ist in jedem Smartphone ein GPS-Empfänger, welcher die Position mit 1Hz aktualisiert[29, 31, S. 1ff]. In Smartphones wie dem Samsung Galaxy S5[30] sind drei unterschiedliche GPS-Empfänger[2] integriert, sodass die Aktualisierung zwischen 1Hz und 3Hz liegt. Weiterhin existieren die ersten Empfänger, die mit 20Hz aktualisieren[18]. Ein GPS System, welches global navigiert, kann global eingesetzt werden, sodass die Einsatzfähig unabhängig vom Standort auf der Erde ist[3], dennoch ist eine freie Sicht vom Anwender zu mehreren Satelliten nötig um eine genaue Ortung zu gewährleisten, sodass in Städten durch große Gebäude, oder Tunnel die Leistungsfähigkeit von GPS beeinträchtigt ist. Basierend auf dieser Argumentation erfüllt ein GPS System die Anforderungen 6,7 und 8 im Abschnitt 4.4.

[2] NAVSTAR, GLONASS und Baidu
[3] GPS Systeme wie das Quasi-Zenith Satellite System ist nur innerhalb Japans Verfügbar[63].

5.1.3 Auswahl

MLS ist nicht weltweit verfügbar, weil es Infrastruktur abhängig ist, wohingegen GPS Systeme weltweit verfügbar sind, aber durch große Gebäude an Leistungsfähigkeit einbüßen, sodass MLS und GPS sich für die Ortung der Teilnehmer ergänzen. In Städten mit hohen Verkehrsaufkommen ist eine höhere Positionsauflösung und Aktualisierungsgeschwindigkeit nötig, sodass der verwendete GPS Service durch MLS ergänzt wird, wenn eine Internetverbindung möglich ist. Außerhalb von Städten wird das GPS Signal weniger blockiert, wodurch die Qualität steigt. Die Leistungsfähigkeit von MLS sinkt, weil weniger Sendestationen verfügbar sind. Es ergibt sich eine Positionsbestimmung, die durch ein oder einer Kombination mehrerer GPS-Services erreicht und optional durch ein System wie MLS ergänzt wird, sodass in Städten eine Ungenauigkeit der Position von 10cm bis 2m angenommen wird (vgl. [13, S. 22]) und eine Aktualisierungsgeschwindigkeit zwischen 1Hz und 10Hz. Daraus folgt bei einer Geschwindigkeit von $\frac{14m}{s}$ eine Auflösung zwischen 14m bzw. 1,4m möglich ist. Basierend auf dieser Argumentation hält die ausgearbeitete Lokalisierung die Anforderungen 6, 7 und 8 im Abschnitt 4.4 ein.

5.2 Protokoll Stack

Dieses Kapitel soll die Auswahl des *Protokoll Stacks* argumentieren, auf Basis der Anforderungen im Kapitel 4.4, die durch das Szenario und den Anwendungsbereich definiert sind. Folgend soll für jeden Layer diskutiert werden welches Protokoll welchen Vorteil für das Assistenzsystem bereitstellt und wie kompatibel es mit den Anforderungen ist.

5.2.1 Physical Layer

Der Physical Layer ist der Layer 1 im OSI-Modell (vgl. 2.1) und stellt die Schnittstelle zum Übertragungsmedium dar. Das Modulationsverfahren definiert wie das Datensignal mit einem Trägersignal auf das Medium moduliert wird. Dabei kann das Modulationsverfahren an das Szenario angepasst sein, sodass der Kanal effizienter genutzt wird und ein Vorteil für das Assistenzsystem entsteht, welches im folgendem Abschnitt erörtert wird.

OFDM

Orthogonal Frequency Division Multiplexing (OFDM) ist ein Modulationsverfahren auf dem Physical Layer, bei dem das Datensignal gleichzeitig auf orthogonale Trägerfrequen-

zen moduliert wird, um das Übersprechen der Signale zu minimieren (siehe 2.2.1). Das Datensignal wird in parallele Datenströme aufgeteilt und auf einen seperaten Träger moduliert, wodurch die Datenrate abnimmt, die Zeitdauer pro Symbol steigt und dadurch wird die Robustheit von OFDM Signalen gegenüber Signalübersprechen erhöht[61]. Weiterhin wird durch ein Guard Intervall[4] die Inter-Symbol Interference (ISI) minimiert, indem zwischen zwei Symbolen für eine wohldefinierte Zeitdauer nicht gesendet wird. Durch die Orthogonalität der Trägersignale ist die Inter Carrier Interference (ICI) bereits minimiert. Durch ein Guard Intervall und Orthogonale Trägersignale wird die Qualität und Effizienz des Kanals gesteigert, indem weniger Interferenzen zu einem Paketverlust führen. In OFDM Systemen sind Ressourcen in der Zeit- (OFDM Symbole) und Frequenzdomäne (Trägersignale) verfügbar. Die Zeit- und Frequenzressourcen können in Unterkanälen organisiert werden und durch unterschiedliche Benutzer beansprucht werden[61].

5.2.2 MAC Layer

Der MAC Layer definiert die (priorisierte) Zuteilung auf das Übertragungsmedium, wobei in Algorithmen unterschieden wird die auf einer zentralen Sendeeinheit basieren oder dezentralisiert sind. Das Anwendungsszenario ein unstrukturiertes, konstant änderndes Netzwerk aus Automobilen, sodass die Vorteile die durch zentralisierte MAC-Algorithmen wie *TDMA*, *CDMA* und *FDMA* zur Verfügung stehen, von der Voraussetzung nicht umsetzbar sind. Die genannten MAC-Algorithmen basieren auf einer zentralen Sendeeinheit (z.B. einen Access Point AP), welche die Ressource zuweist, wie einen Frequenz- und Zeitslot, um innerhalb diesen zu senden (FDMA) (vgl. [52]). Folgend wird das CSMA/CA und STDMA MAC-Protokoll diskutiert, da ersteres das Standard MAC Protokoll im IEEE 802.11p (vgl. 2.3) und kanalbasiert ist, welches für den Anwendungsfall der Inter-Vehicle Communication (IVC) standardisiert ist und das letztere welches ein dezentrales, paketbasiertes TDMA Protokoll ist und auf den Koordinaten der Teilnehmer basiert um Slots neu zu belegen. Die Auswahl für das STDMA Protokoll ist beeinflusst durch die Verfügbarkeit der Koordinaten für das Assistenzsystem und der Implementation in der Luftfahrttechnik um Daten zwischen Flugzeugen und Basisstationen auszutauschen (VHF Data Link (VDL) mode 4 System[52])

[4] Ein Zeitlicher oder frequenzieller Abstand zwischen zwei distinkten Symbolen bzw. Kanälen um Interferenzen zu minimieren.

CSMA/CA

CSMA/CA ist eine Non-Persistente Erweiterung von CSMA, indem durch Abhören des Übertragungsmediums für ein AIFS und anschließenden senden, wenn das Medium frei ist, bzw. warten durch den Backoff-Algorithmus, wenn das Medium belegt ist, Kollision auf dem Medium verhindert werden (vgl. 2.2.2). Diese Vorgehensweise im CSMA/CA Protokoll kann zu unbegrenztem warten führen[52], sodass in einer zeitkritischen Anwendung 80% der Nachrichten verworfen werden[5], da kein Zugriff auf das Übertragungsmedium erlangt wird (vgl. [52]), weil keine feste Worst-Case-Zugriffszeit existiert[52].

STDMA

STDMA ist eine Verbesserung von TDMA um eine selbstorganisierte Zuweisung von Sende Slots innerhalb eines Frames auf dem Übertragungsmediums (vgl. 2.2.2). TDMA ist ein zentralisierter MAC-Algorithmus, wohingegen STDMA dezentralisiert ist und für diesen Anwendungsfall in Betracht gezogen wird (vgl. 2.2.2). STDMA basiert auf den Koordinaten der einzelnen Teilnehmer, wodurch Slots neu zugeordnet werden, wenn das Übertragungsmedium ausgelastet ist, sodass die Slots von Teilnehmern, die am weistesten entfernt sind, neu Belegt werden (vgl. 2.2.2) und eine feste Worst-Case-Zugriffszeit von Abhören des Übertragungsmediums plus Nominal Increment möglich ist (vgl. 2.2.2, [52]). Weiterhin ist das neu belegen von Slots nach einer zufälligen Zahl zwischen 3 und 8 Durchläufen definiert, wodurch das Protokoll sich den Bedingungen eines dynamischen Systems anpasst[52].

EDCA

EDCA ist ein MAC Protokoll, welches die Zuteilung des Übertragungsmediums durch Access Categories (AC) 0-3 priorisiert (vgl. 2.2.2). Priorisierte Pakete warten für ein kürzeres AIFS und werden schneller verschickt. Daraus folgt, dass priorisierte Nachrichten wie Safety oder Heartbeat früher versendet werden als einfache Update-Nachrichten.

5.2.3 Höhere Layer

Für die Layer 3 und 4 ist eine Kombination der Protokolle IP/TCP und IP/UDP (siehe 2.2.3 und 2.2.4) möglich. IP bietet die Möglichkeit Sender und Empfänger logisch zu adressieren, oder das zu sendende Paket als Broadcast zu markieren[6], wodurch jeder

[5] Bei 1000m Sendebereich, 10Hz Updateinterval, 500 Byte großen Nachrichten und 230 Sendern[52]
[6] Durch eine IP-Adresse die für Broadcast Nachrichten reserviert ist.

Empfänger entscheidet, ob eine empfangene Broadcast Nachricht bearbeitet oder verworfen wird. TCP ist ein verbindungsorientiertes Transportprotokoll mit Congestion Control (siehe 2.2.4), welches bei einem Paketverlust von einen Stau auf dem Übertragungsmedium ausgeht, weil es für drahtgebundene Netzwerke konzipiert ist, beidem die Bit-Error Fehler gering sind[19]. Das beschriebene Verhalten ist für C2X Anwendungen ungeeignet[19], da das Szenario für C2X Anwendungen dynamische, stetig ändernde Systeme sind, bei dem versendete Nachrichten den Zustand des Netzwerkes zu einen definierten Zeitpunkt repräsentieren. Die genannten Nachteile von TCP sind in UDP nicht vorhanden, weil es verbindungslos ist und kein Congestion Control implementiert, sodass UDP einen besseren Datendurchsatz als TCP hat und TCP ein besseres Delivery Ratio[7] als UDP aufweist[20]. Auf Basis dieser Argumentation ist IP/UDP den Protokollen IP/TCP überlegen, dennoch bieten IP/UDP keine zusätzlichen Funktionalitäten, die oberhalb von Layer 1 und 2 benötigt werden und sind nicht weiter berücksichtigt.

5.2.4 Auswahl

Durch OFDM wird die Bandbreite im Kanal effizient ausgenutzt und die propagierten Signale sind durch einen längeren Symbol Sendezyklus robuster gegen Signalübersprechungen [61], welche durch Autos und Hauswände im Szenario hervorgerufen wird. Basierend auf dieser Argumentation wird für die Luftschnitstelle die OFDM Modulationstechnik verwendet und die Anforderungen 9 und 12 im Kapitel 4.4 eingehalten.

Auf dem Layer zwei als MAC Algorithmus soll das STDMA Protokoll verwendet werden, da es eine feste obere Grenze für die Kanalzugriffszeit definiert[52], welche für Safety Anwendungen vorausgesetzt wird und der Anforderung 11 im Kapitel 4.4 entspricht. Die feste obere Grenze für den Kanalzugriff wird implementiert durch die verfügbaren Koordinaten der Verkehrsteilnehmer, die durch das Assistenzsystem gegeben sind, und dem Übersenden vom Slots, dessen Inhaber am entferntesten ist[52]. Im STDMA ist ein Guard Intervall zwischen den Slots implementiert, wodurch ein Übersprechen benachbarter Slots[8] minimiert wird (Anforderung 12), wohingegen bei CSMA das Medium für ein AIFS (vgl. 2.2.2) abgehört wird und bei freiem Medium direkt gesendet wird, welches ein Übersprechen unterschiedlicher Signale nicht ausschließt, sodass keine feste obere Grenze für den Kanalzugriff existiert[52]. In einem direkten Vergleich mit einem 500 Byte großem Paket, einer Inter-Arrival Zeit[9] von 100ms, einer Aktualisierungsrate von

[7] $DeliveryRatio = \frac{Fehlerfrei\ Empfangene\ Pakete}{Gesendete\ Pakete}$
[8] Z.B. durch den Doppler-Effekt, da die Kommunikationspartner in Bewegung sind.
[9] Die Zeit zwischen zwei distinkten Paketen.

10Hz, 230 Sender und einem Sendebereich von 1000m können im STDMA Protokoll 30% aller Slots neu zugeordnet[10] werden ohne zusätzliche Paketverluste durch Interferenzen[52] (Anforderungen 11 und 12), wohingegen mit dem CSMA Algorithmus 53% Paketverluste aufgetreten[52]. Basierend auf dieser Argumentation weist das STDMA Protokoll eine größere Leistungsfähigkeit als CSMA/CA auf, sodass eine Verwendung von STDMA favorisiert ist. Wie im Kapitel 4.3 bereits ausgearbeitet, ist das INET Framework (vgl. 2.9) eine OMNeT++ (vgl. 2.5) Erweiterung um Kommunikationsprotokolle und wird kontinuierlich durch die OMNeT++ Community weiterentwickelt, sodass die Veins kompatible INET Version 2.3 nicht das STDMA Protokoll bereitstellt, sodass unter der Berücksichtigung des Leistungsunterschieds dennoch das CSMA/CA Protokoll verwendet wird.

EDCA wird auf dem MAC-Layer verwendet um die Access Categories AC0-3 pro Kanal zu definieren, um einen priorisierten Zugriff auf das Übertragungsmedium zu erhalten, was für Safety Anwendungen vorteilhaft ist (vgl. 2.2.2 und Anforderung 12 4.4).

OFDM, CSMA/CA und EDCA sind für IVC/C2X Anwendungen zusammen als IEEE 802.11p standardisiert (vgl. 2.3). Die Bandbreite für jeden Kanal wird durch das Protokoll P auf 10Mhz halbiert (vgl. 2.3, [48]), verglichen mit dem Standard A, sodass der IEEE Standard P sich dem Anwendungsszenario von C2X anpasst. Die Symboldauer ist verdoppelt und dadurch robuster gegenüber Signalübersprechungen [48] (Anforderung 12). Mit der Verwendung von OFDM, CSMA/CA und EDCA als Protokoll Stack wird von den Anforderungen 9-12 an den Protokoll Stack nur die Anforderungen 9 und 12 eingehalten.

5.3 Assistenzsystem

In diesem Kapitel wird der Simulationsablauf dargestellt und der Aufbau des Assistenzsystems durch Selfmessages konzeptionell ausgearbeitet. Das Fahrverhalten wird definiert und es wird ausgearbeitet, wie das Assistenzsystem Einfluss darauf hat. Für die Teilbereiche *Entfernungsberechnung*, *Erkennung von potenziellen Kollisionen* und *Beschleunigungsverhalten* im Assistenzsystem wird die Funktionsweise konzeptionell ausgearbeitet und die Implementation im Kapitel Implementierung 6 im Detail vorgestellt. Abschließend wird der Aufbau der Broadcast Nachricht dargestellt.

[10] Die durchschnittliche Distanz zwischen zwei Sendern, die im selben Slot funken, ist 825m[52]

5.3.1 Simulationsablauf

SUMO führt die Straßenverkehrssimulation durch, indem die Automobile, wie im *vType* definiert, in die Simulation eingefügt werden und für jedes Automobil, für jeden Simulationsschritt die nächste Position, in Abhängigkeit der im *vType* definierten physikalischen Eigenschaften, berechnet wird. An dem Endpunkt der definierten Route werden die Fahrzeuge aus der Simulation entfernt. Über TraCI ist Veins mit SUMO verbunden, wodurch die Simulationsobjekte und ihre physikalischen Eigenschaften in OMNeT++ anforderbar sind und von OMNeT++ über TraCI das Fahrverhalten der Simulationsobjekte in SUMO manipulierbar ist.

Für jeden Simulationsschritt führt OMNeT++ die Zustellung von Selfmessages und anderen Nachrichten bzw. allgemeiner Ereignissen durch und alsbald alle Ereignisse und zugehörigen Berechnungen abgeschlossen sind, wird SUMO aufgefordert den nächsten Simulationsschritt durchzuführen. Durch das Assistenzsystem wird in Zeitintervallen, wie im Kapitel 4.6.1 definiert, ein Update Broadcast versendet, welches über das Veins Framework an alle Automobile versendet wird durch die *Dynamic Connection Creation*[45, Kapitel 4.13.6] in OMNeT++. Auf Basis der Eigenschaften der Fahrzeuge, wie Koordinaten und Abstand, zum sendenden Fahrzeug und anderer Signale, die im Simulationsschritt zu senden sind, wird durch Veins die Interferenz berechnet und der Broadcast fehlerfrei oder mit Bit-Error zugestellt.

Der Update Broadcast wird von allen Verkehrsteilnehmern generiert und versendet. Alle Verkehrsteilnehmer mit Assistenzsystem führen auf Grundlage der empfangenen Nachrichten die Berechnungen für die Kollisionsvermeidung durch. Der Aufbau des Update Broadcasts ist in Tabelle 5.1 einzusehen.

5.3.2 Verhalten der Fahrzeuge in der Simulation

Fahrzeuge werden wie im Kapitel 4.6.4 beschrieben in die Simulation eingefügt. Nach dem Einfügen kann jedes Fahrzeug mit jedem anderen eine Kollision verursachen. Eine Kollision ist definiert als Berührung von zwei Fahrzeugen. Fahrzeuge beschleunigen und folgen dem Straßenverlauf nach der Formel im Listing 5.1, wenn das Automobil keinem anderen Automobil folgt.

```
1 min(max(Speed of Street)*speedFactor, max(Next Speed), (max Speed
      of Vehicle))
```

Listing 5.1: Beschleunigung der Fahrzeuge durch das Car Following Modell.

max(Speed of Street) beschreibt die maximal erlaubte Geschwindigkeit, auf der jeweiligen Straße, multipliziert mit dem *Speed Faktor* aus der Standardverteilung, definiert in der *vType* Definition der Fahrzeugtypen. *max(Next Speed)* beschreibt die maximale Geschwindigkeit, die in Abhängigkeit von der aktuellen Geschwindigkeit und maximaler Beschleunigung möglich ist. *max Speed of Vehicle* ist die im *vType* definierte maximale Geschwindigkeit des jeweiligen Fahrzeugtyps. Wenn ein Automobil hinter einem anderen fährt, dann ändert sich das Verhalten und es wird in zwei Fälle unterschieden.

1. Die Beschleunigung des vorausfahrenden Fahrzeug ist $a \geq 0$.
   - Der Fahrer reagiert innerhalb einer Reaktionszeit auf eine Anpassung an den Mindestabstand (vgl. Tabelle 4.1).
2. Die Beschleunigung des vorausfahrenden Fahrzeug ist $a < 0$.
   - Das vorausfahrende Fahrzeug bremst. Das wird durch das Assistenzsystem erkannt und alsbald dem Fahrer signalisiert. Nach einer Reaktionszeit reagiert der Fahrer auf die Signalisierung durch das Assistenzsystem und bremst (vgl. Tabelle 4.1).

Das Fahrverhalten ändert sich nur durch eine aktive Modifikation, oder durch die Übergabe der Kontrolle an SUMO. Hat das Assistenzsystem eine potenzielle Kollision erkannt, dann wird dem Fahrer signalisiert zu bremsen. Der Fahrer setzt den Befehl nach einer Reaktionszeit um.

5.3.3 Selfmessages

Jedes Automobil implementiert OMNeT++ Selfmessages (vgl. 2.5.4). Die Fahrzeuge mit Assistenzsystem implementieren *SimView*, *Driver*, *Update Broadcast* und *CrashVoid*. Ausnahmefahrer implementieren *Driver* und *CrashVoid*.

1. **SimView**

   Die SimView Selfmessage fasst die Koordinaten aller Szenarioteilnehmer zusammen, indem in jedem Simulationsschritt die Koordinaten von SUMO angefordert werden und zur Verfügung gestellt werden. Im zweiten Teil wird auf Basis der Koordinaten die Automobile analsiert und protokolliert, wenn eine Kollision entsteht.

2. **Driver**

Die Driver Selfmessage startet den Algorithmus, der auf Basis der durch SimView erlangten Koordinaten das Automobil durch den Straßenverkehr alle *Reaktionszeit* Sekunden bewegt (vgl. 4.6.1). Dieses Verhalten wird durch den Algorithmus CrashVoid manipuliert, wenn eine Kollision erkannt wird.

3. **Update Broadcast**

   Der Update Broadcast, mit dem Aufbau in Tabelle 5.1, wird in Abhängigkeit von der Iterationsvariable *Sendeinterval* (vgl. 4.6.1) erstellt und über das Veins Framework zugestellt.

4. **CrashVoid**

   Die CrashVoid Selfmessage startet den Algorithmus für die Kollisionserkennung und informiert den Driver Algorithmus über die Ergebnisse durch eine Queue (Anforderung 15). Die n-te Algorithmus Selfmessage löst alle $Simulationsschrittlaenge + n_{-1} - te\ Algorithmuslaufzeit$ Sekunden aus. Für Standardfahrzeuge wird jede erkannte Kollision übermittelt und für Ausnahmefahrer ausschließlich für das Auffahrszenario.

5.3.4 Analyse der Empfangenen Nachrichten

Die empfangen Broadcasts der Verkehrsteilnehmer werden analysiert und gefiltert, sodass die Effizienz gesteigert wird. Automobile, mit denen eine Interaktion ausgeschlossen ist, werden für den weiteren Verlauf des Assistenzsystem nicht beachtet. Auf Basis der Fahrrichtung und Koordinaten wird festgestellt, ob ein Automobil hinter, vor dem betrachteten fährt, oder die Fahrbahn gekreuzt. Verkehrsteilnehmer, die hinter dem betrachteten fahren und Fahrzeuge, die den CP überfahren haben und der Abstand zum CP ausreichend groß ist, werden für die weitere Bearbeitung nicht beachtet.

5.3.5 Unfallvermeidung

Die Unfallvermeidung wird implementiert durch die *Entfernungsberechnung*, *Erkennung von potenziellen Kollisionen* und dem *Beschleunigungsverhalten*. Die *Entfernungsbestimmung* bearbeitet das Problem der Positionsungenauigkeit, wenn die Distanz zwischen zwei Punkten berechnet wird und muss die Anforderung 13 einhalten. Der Abschnitt *Beschleunigungsverhalten* befasst sich mit der Berechnung von einer neuen Geschwindigkeit, wenn auf ein vorausfahrendes Fahrzeug reagiert wird, oder an der Kreuzung eine Kollision zu vermeiden ist. Das Konzept für das Beschleunigunsverhalten muss die

Anforderung 14 im Kapitel 4.4 einhalten. In dem Abschnitt *Erkennung von potenziellen Kollisionen* wird das Konzept erarbeitet mit dem zukünftige Kollisionen erkannt werden für das Kreuzungs- und Auffahrszenario (Anforderung 15).

Entfernungsberechnung

In diesem Abschnitt wird das Konzept für die Anforderung 13 im Kapitel 4.4 ausgearbeitet. Für die Anforderung 13 ist in die zwei Fälle Auffahrszenario und Kreuzungsszenario zu unterscheiden, weil beide Szenarien ein unterschiedliches Verhalten bezüglich der Positionsungenauigkeit aufweisen. Für das Auffahrszenario ist zu erkennen, dass die Positionsungenauigkeit eindimensional auftritt, wenn der Verkehrsteilnehmer als vorausfahrend festgestellt ist, wohingegen im Kreuzungsszenario die Positionsungenauigkeit zweidimensional auftritt. Im Auffahr- und Kreuzungsszenario kann durch eine Abschätzung der maximalen Näherung die Anforderung 13 eingehalten werden. Von der berechneten Distanz zum vorausfahrenden Automobil wird die Hälfte der kombinierten Positionsungenauigkeit subtrahiert, sodass der Abstand zwischen den Automobilen ausreichend groß abgeschätzt ist. Für das Kreuzungsszenario wird von der berechneten Entfernung zum Kollisionspunkt (CP) die vollständige Positionsungenauigkeit subtrahiert, sodass für das zweidimensionale Eintreten der Positionsungenauigkeit beide Verkehrsteilnehmer bedacht sind.

Erkennung einer potenziellen Kollisionen

Für die Erkennung von potenziellen Kollisionen werden die Koordinaten aus den Broadcasts der Verkehrsteilnehmer analysiert, sodass die Anforderung 15 eingehalten wird. Zukünftige Positionen werden durch verfügbare Koordinaten approximiert, sodass festgestellt wird, ob die Fahrbahnen der Verkehrsteilnehmer Schnittpunkte (CP) aufweisen. Ist für ein Kreuzungsszenario ein Schnittpunkt der Fahrbahnen erkannt, dann wird überprüft, ob dieser für beide Verkehrsteilnehmer voraus liegt, und, ob beide Verkehrsteilnehmer im selben Zeitintervall den CP durchfahren. Sind beide Bedingungen erfüllt, dann liegt eine potenzielle Kollision voraus. Für das Auffahrszenario ist durch Fahrrichtung und Koordinaten bekannt welche Automobile vorausfahren, sodass auf diese in Abhängigkeit von Abstand und Bremsung zu reagieren ist.

Beschleunigungsverhalten

In diesem Abschnitt wird das Konzept für die Anforderung 14 im Kapitel 4.4 ausgearbeitet. Die Anforderung 15 beschreibt das Beschleunigungsverhalten der Verkehrsteilnehmer, wenn eine Kollision erkannt wird. Für das Beschleunigungsverhalten wird in das Kreuzungs- und Auffahrszenario unterschieden. Für das Auffahrszenario tritt die Beschleunigung positiv und negativ auf. Es wird überprüft, ob der Abstand kleiner als der Gefahrenabstand von zwei Metern ist. Ist der Abstand kleiner als der Gefahrenabstand wird die Geschwindigkeit auf null gesetzt. Ist der Abstand größer als der Gefahrenabstand wird die neue Geschwindigkeit durch das Verhältnis von Soll- und Haben Abstand ermittelt. Der aktuelle Abstand in Sekunden wird durch den *Mindestabstand* dividiert und mit der aktuellen Geschwindigkeit multipliziert und entspricht der neuen Geschwindigkeit. Für das Kreuzungsszenario tritt die Beschleunigung ausschließlich negativ auf. Es muss festgestellt sein, dass eine Schnittmenge für die Zeit auf dem CP von zwei Automobilen vorliegt. Ist diese Schnittmenge vorhanden, dann wird eine neue sichere Zeit zum CP berechnet. Die zusätzliche Zeit, die bis zum CP benötigt wird, berechnet sich durch die Summe aus Gefahrenabstand, Fahrzeugbreite, Länge und *Positionsungenauigkeit* dividiert durch die Geschwindigkeit des anderen Fahrzeuges. Durch die Addition der zusätzlich benötigten Zeit zur aktuellen Fahrzeit wird die neue Fahrzeit berechnet. Mit dieser Rechnung ist versichert, dass der andere Fahrzeugführer bei gleichbleibender Geschwindigkeit ausreichend Zeit hat, um die Kreuzung kollisionsfrei zu überqueren.

5.3.6 Nachrichtenaufbau

In der Tabelle 5.1 ist der Aufbau der Nachricht für den Broadcast Update zu sehen.

Die Variable *NodeId* wird benötigt um eine Menge von Nachrichten einem konkreten Verkehrsteilnehmer zuzuordnen und dadurch das Fahrverhalten zu untersuchen. Der *TimeStamp* repräsentiert die Zeit, die durch den Lokalisierungsservice gegeben und somit für alle Verkehrsteilnehmer identisch ist. *SenderPos* gibt die berechneten Koordinaten zum Zeitpunkt *TimeStamp* mit einer optionalen Ungenauigkeit an, woraus im folgenden die Fahrrichtung, Geschwindigkeit und Beschleunigung abgeleitet wird, sodass keine Ungenauigkeit einzurechnen sind. Die *Direction* gibt an in welche Richtung sich der Sender bewegt, sodass mit Hilfe der Fahrrichtung das Fahrverhalten der Verkehrsteilnehmer effizienter analysiert wird. *Geschwindigkeit* und *Beschleunigung* geben die aktuelle Geschwindigkeit und Beschleunigung des Automobils an. *CarLength* gibt die Fahrzeuglänge des Senders an und hat damit einen Einfluss wie stark für einen Verkehrsteilnehmer

Datentyp	Name	Funktion
Integer	CarId	CarId stellt eine eindeutige Identifikation der Verkehrsteilnehmer dar.
Double	Direction	Direction entspricht der Richtung in Rad, in die das Automobil fährt.
Double	CarLength	Entspricht der Länge des Automobils.
3 Double	SenderPos	Die Koordinaten des Senders im dreidimensionalen Raum.
Double	TimeStamp	Zeitstempel zu dem Zeitpunkt der Nachrichtenerstellung.
Double	Geschwindigkeit	Aktuelle Geschwindigkeit des Senders.
Double	Beschleunigung	Aktuelle Beschleunigung des Senders.

Tabelle 5.1: Aufbau der Broadcast Nachricht für das Update.

gebremst wird, wenn ein gemeinsamer Kollisionspunkt vorhanden ist. Wie in Tabelle 5.1 zu erkennen ist, ist der Broadcast 68 Byte und mit dem 32 Byte Header 100 Byte groß.

5.4 Statistik

Dieser Abschnitt zeigt, wie die Signale aus dem Arbeitspaket 5 im Abschnitt 4.5 berechnet werden. Die Statistiken werden wie im Kapitel 2.5.5 erzeugt um die Hypothesen im Kapitel 4.7 zu belegen und die Eigenschaften des Assistenzsystems zu untersuchen. Folgend wird dargestellt aus welchen Daten die Signale bestehen und was durch die Signale gezeigt wird.

a) Um die Eigenschaften des Szenarios zu interpretieren wird durch Veins die Zeit im Szenario (TT) und die gefahrene Strecke (TD) für jedes Automobil dokumentiert.

b) Für die Hypothese 1 wird die Summe der Empfangenen Broadcasts (RBC) und Broadcasts mit Bit-Error (TLP) benötigt, die durch Veins dokumentiert wird.

c) Für die Untersuchung des MAC-Algorithmus dokumentiert Veins für jeden Teilnehmer die Summe der Backoff Durchläufe (TIB).

d) Für die Untersuchung des MAC-Algorithmus dokumentiert Veins für jeden Verkehrsteilnehmer die Summe der im Backoff-Algorithmus gewarteten Slots (WS).

e) Für die Untersuchung des MAC-Algorithmus dokumentiert Veins für jeden Teilnehmer die Summe der gesendeten Broadcasts (TS).

f) Für die Hypothesen 1-3 wird die Zeit zu einem Kollisionspunkt (T2CP) betrachtet in Abhängigkeit der Veränderung vom *Sendeintervall*, *Mindestabstand* und *Positionsungenauigkeit* (vgl. 4.6.1). In der Simulation wird erfasst wie viel Zeit bis zum Kollisionspunkt (CP) verbleibt, wenn eine Kollision auf Basis der aktuellen Geschwindigkeit und Abstand zum CP erkannt wird.

g) Für die Hypothesen 1-5 wird die Summe der Kollisionen (*Kollisionen*) in Abhängigkeit der Veränderung vom *Sendeintervall*, *Mindestabstand*, *Positionsungenauigkeit*, *Ausnahmefahrer* und Reaktionszeit (vgl. 4.6.1) betrachtet. Es wird dokumentiert, wenn eine Kollision zwischen zwei Verkehrsteilnehmern eingetreten ist.

Für die Statistiken *RBC, TLP, TIB, WS, TS* und *Kollisionen* wird die Statistik mit dem Filter *last(sum)* erstellt, für die Statistiken *T2CP* mit dem Filter *last(mean)* und für *TT* und *TD* mit *last()*(siehe 2.5.5). Die Ergebnisse der Statistiken *T2CP, Kollisionen, RBC, TLP, TIB, WS, TT, TD* und *TS* werden über die Anzahl der Simulationswiederholungen gemittelt. Die Summe der quadrierten Differenzen von dem Durchschnitt und allen Stichproben dividiert durch die Anzahl der Stichproben heißt Varianz. Die Varianz beschreibt wie verstreut die Stichproben sind, wobei mit kleiner Varianz die Stichproben eng beieinander liegen und vice versa. Die Quadratwurzel der Varianz heißt Standardabweichung und beschreibt den durchschnittlichen Abstand der Stichproben zum Erwartungswert, woraus folgt, dass innerhalb einer Standardabweichung 68,3% aller Stichproben liegen.

Jedes erstellte Diagramm zeigt eine oder mehrere Statistiken in Abhängigkeit einer Iterationsvariable. Zusammen mit jedem Diagramm wird eine Tabelle gezeigt die für jede Statistik die Stichprobengröße, Erwartungswert, Standardabweichung und Konfidenzintervall zeigt, wodurch die Qualität und Eigenschaften der Statistik deutbar ist. Für diese Ausarbeitung ist ein Konfidenzintervall von 95% gewählt und wird berechnet indem für den Wert 0,95 aus einer Z-Tabelle[11] der zugehörige Faktor bestimmt wird, mit der die Standardabweichung dividiert durch die Quadratwurzel der Stichprobengröße multipliziert wird. Für einen Durchschnitt x sagt das Konfidenzintervall y aus, dass der wahre Wert zu 95% in dem Intervall $[x - y, x + y]$ enthalten ist.

Im Kapitel Evaluierung 7 werden die drei dargestellten Teilbereiche *Szenario*, *Übertragungsmedium* und *Leistungsfähigkeit des Assistenzsystem* untersucht. Für das *Szenario*

[11] Quelle http://eswf.uni-koeln.de/glossar/zvert.htm

wird mit den Statistiken *TT* und *TD* ausgearbeitet wie schnell sich die Fahrzeuge innerhalb einer Standardabweichung bewegen. Anschließend wird das Ergebnis interpretiert und die Auswirkung auf die folgende Ausarbeitung dargestellt. Für das *Übertragungsmedium* werden die Statistiken *TLP*, *RBC*, *TD*, *WS* und *TS* in Abhängigkeit vom *Sendeintervall* dargestellt um die Frage zu klären wie groß der Einfluss des MAC-Algorithmus auf die effiziente Kanalnutzung und Zugriff auf das Übertragungsmedium ist. Der Abschnitt *Leistungsfähigkeit des Assistenzsystem* teilt sich in die zwei Bereiche für *T2CP* und *Kollisionen* auf. Im Abschnitt *T2CP* wird untersucht wie sich die Iterationsvariablen *Sendeintervall*, *Mindestabstand* und *Positionsungenauigkeit* auf die Leistungsfähigkeit der Kollisionsfrüherkennung auswirken. Im Abschnitt *Kollisionen* sind die Iterationsvariablen *Sendeintervall*, *Mindestabstand* und *Positionsungenauigkeit*, wie für *T2CP*, untersucht um die Auswirkung auf das Entstehen von Kollisionen zu untersuchen und diese Ergebnisse durch die Diagramme aus dem Abschnitt *T2CP* zusätzlich zu bestätigen, wenn die gleiche Korrelation zu erkennen ist. Außerdem wird die Statistik *Kollisionen* in Abhängigkeit von den Iterationsvariablen *Reaktionszeit* und *Ausnahmefahrer* betrachtet, die Unabhängig von *T2CP* sind und von Außerhalb durch den Fahrer -*Reaktionszeit*- und durch das Szenario -Anzahl der Verkehrsteilnehmer ohne Assistenzsystem- gegeben sind.

6 Implementierung

Dieses Kapitel beschäftigt sich mit der Implementierung der Arbeitspakete, welche im Kapitel 4.5 definiert sind. Der Abschnitt 6.1 beschreibt den Versuchsaufbau der Simulation. Für SUMO wird erklärt, welche Dateien für eine Simulation benötigt und wie diese generiert werden. Weiterhin wird aufgezeigt, wie Veins mit dem OSM und 2RIM initialisiert wird. Im Abschnitt 6.2 werden die Arbeitpakete 1 und 2 bearbeitet. Es wird gezeigt mit welchen Methoden das *TraCICommandInterface* im Veins Framework erweitert ist und wie der Speed Modi für die Automobile in der SUMO Simulation angepasst wird. Im Kapitel 6.3 werden die Arbeitspakete 3 und 4 bearbeitet. Das Kapitel zeigt, wie das Assistenzsystem und das CFM in Veins implementiert sind. Das Assistenzsystem und das CFM sind durch die Selfmessages (vgl. 5.3.3) CrashVoid und Driver implementiert und die statistische Auswertung durch SimView. Dieses Kapitel beschäftigt sich mit der Logik, die durch *CrashVoid*, *Driver* und *SimView* implementiert ist und erläutert diese mit Ablaufdiagrammen.

6.1 Versuchsaufbau

Wie im Kapitel 2.8 beschrieben, wird der SUMO Server, für die Kommunikation zwischen Veins und SUMO, mit dem Befehl `python $HOME/veins-4.4/sumo-launchd.py -vv {-c sumo -gui}` gestartet. Durch den SUMO Server ist ein Zugriff auf die Verkehrssimulation möglich. Anschließend wird eine SUMO Simulation über das TraCI Interface gestartet. In diesem Abschnitt wird erläutert wie die SUMO und Veins Simulation initialisiert sind. Zuerst wird dargestellt, was Netzwerk-, Polygon- und Route Files sind und wie sie für eine SUMO Simulation generiert werden. Anschließend wird dargestellt, wie das OSM und 2RIM (vgl. 2.7.1 und 2.7.1) in die Simulation eingebunden worden.

6.1.1 SUMO Initialisierung

Mit dem Befehl `sumo{-gui} -c {name}.sumo.cfg` wird eine SUMO Simulation gestartet, mit dem Network-, Route-, Polygon Dateien, die in *{name}.sumo.cfg* angegeben sind.

Weiterhin wird die Simulationsschrittlänge im cfg File festgelegt, mit der Veins und SUMO den Simulationsfortschritt berechnen und durchführen. Für diese Ausarbeitung wird eine Schrittlänge von 100ms gewählt.

Das Network File wird mit dem SUMO Tool NETCONVERT erstellt, indem aus Open Street Map[1] ein Szenario ausgewählt und als Datei mit der Endung.osm geladen wird. Eine OSM Datei beinhaltet für den ausgewählten Bereich eine detaillierte Auflistung der Eigenschaften, die den Ausschnitt repräsentieren. Mit dem Befehl `netconvert --osm {name}.osm -o {name}.net.xml` wird das OSM Datenformat in das SUMO Network Format übersetzt und von SUMO Werkzeugen verstanden. Durch Straßen wird, mit einer maximalen Geschwindigkeit, Länge, Form und Verbindung zueinander ein Szenario beschrieben. Im Network sind die Straßen beschrieben, auf denen die Automobile in der Simulation fahren.

Veins berechnet aus den Daten in der Polygon-Datei die physikalischen Einflüsse auf die Signalausbreitung. Die Polygon-Datei wird mit dem SUMO Tool POLYCONVERT erstellt. Mit dem Befehl `polyconvert --osm {name}.osm -o {name}.poly.xml` wird aus der *{name}.osm* die Gebäudeinformationen in das Format *{name}.poly.xml* überführt und ist nun lesbar für SUMO Anwendungen. In der Polygon-Datei sind Gebäude und Mauern, Positionen und geometrische Eigenschaften beschrieben.

Die Route-Datei beinhaltet die vType und Route-Definitionen der Automobile (vgl. 2.6.2, 4.6.3 und 4.6.2). In der vType und der Route-Definition sind Flows definiert (vgl. 2.6.2).

Auf Basis der Network-, Polygon- und Route-Dateien startet SUMO eine Simulation, in der sich Automobile wie im Route-File definiert durch ein Szenario bewegen, welches durch die Network- und Polygon-Dateien beschrieben ist.

6.1.2 Veins Initialisierung

Die im Abschnitt (vgl. 2.7.1 und 2.7.1) dargestellten Modelle für das Übertragungsmedium werden in die Simulation integriert, indem der Physical Layer in Veins mit den Einstellung der XML-Datei aus dem folgenden Listing 6.1 in der OMNeT++ *init* Datei initialisiert wird.

```
1 <root>
2  <AnalogueModels>
3   <AnalogueModel type="TwoRayInterferenceModel">
4    <parameter name="DielectricConstant" type="double"
```

[1] https://www.openstreetmap.de/

```
                value="1.02"/>
  </AnalogueModel>
  <AnalogueModel type="SimpleObstacleShadowing">
   <parameter name="carrierFrequency" type="double"
                value="5.890e+9"/>
  </AnalogueModel>
 </AnalogueModels>
</root>
```

Listing 6.1: Einstellungen für den physical Layer in Veins.

Im Listing 6.1 wird Veins mitgeteilt, dass das TRIM und OSM anzuwenden ist. Im 2RIM wird eine dielektrische Konstante von 1.02 bestimmt, welche die Brechungsstärke definiert, die Signale erfahren, wenn sie Nichtleiter durchdringen. In dem OSM wird die mittlere Sendefrequenz von 5,89GHz bekannt gegeben.

6.2 Modifikation von Veins und SUMO

In diesem Abschnitt werden die Arbeitspakete 1a und 2a bearbeitet, welche im Kapitel 4.5 definiert sind.

6.2.1 Veins Erweiterung

Das erste Arbeitspaket befasst sich mit der Veins Erweiterung, wodurch jedes Automobil kontextunabhängig Daten von jedem Simulationsobjekt erhält. Dieses Erweiterung wird von den Algorithmen *SimView*, *Driver* und *Update Broadcast* für die Funktionsfähigkeit benötigt. In der Veins Klasse *TraCICommandInterface* sind die Methoden aus Listing 6.2 definiert und implementiert.

```
double getCarLength();
std::list<std::string> getVehicleIds();
Coord getCarPosition(std::string carId);
double getCarLength(std::string carId);
double getCarSpeed(std::string carId);
double getCarDir(std::string carId);
```

Listing 6.2: Veins Erweiterung in der Klasse TraCICommandInerface.

Im Listing 6.2 sind alle Methoden zu sehen, mit denen Veins erweitert ist. Die *getCarLength()* Methode wird in der Initialisierungsphase von jedem Automobil aufgerufen und der Wert gespeichert. Die Fahrzeuglänge wird mit jeder Broadcast Nachricht gesendet.

Die *getVehicleIds()* Methode fordert von SUMO von allen Simulationsobjekten die *carIds* an, welche in den folgenden vier Methoden benötigt wird. In den letzten vier Methoden wird die Position, Länge, Geschwindigkeit und Ausrichtung in Abhängigkeit der *carId* von SUMO angefordert. Auf Basis dieser vier Methoden findet die Fortbewegung der Fahrzeuge durch *Driver* und die statistische Datenerfassung durch den *SimView* Algorithmus statt.

6.2.2 SUMO CFM Modifikation

Für jedes Automobil ist der *speed mode* frei manipulierbar. Der Speed Mode definiert, welche Ressourcen und Eigenschaften dem CFM zur Verfügung steht, um die Simulationsobjekte durch das Szenario zu bewegen. Die Modi sind in der Tabelle 6.1 einzusehen.

Bit	**Bedeutung**
0	Das Automobil hält die sichere Geschwindigkeit ein, bei der keine Kollision entsteht.
1	Das Automobil hält die maximale positive Beschleunigung aus der vType Beschreibung ein.
2	Das Automobil hält die maximale negative Beschleunigung aus der vType Beschreibung ein.
3	Das Automobil reduziert automatisch die Geschwindigkeit, wenn es sich einer Kreuzung nähert.
4	Das Automobil führt eine Vollbremsung aus, damit keine rote Ampel überfahren wird.

Tabelle 6.1: Speed Modi für das CFM.

Bit 0 basiert auf den vType Eigenschaften anderer Fahrzeuge. Bit 3 und 4 basieren auf präventiven Eingreifen des CFMs, um Kollisionen auf der Kreuzung zu vermeiden. Für diese Ausarbeitung soll das CFM ausschließlich auf Bit 2 und 3 basieren und die festgelegten Beschleunigungsgrenzen einhalten. Der Speed Mode wird mit dem Befehl im Listing 6.3 über das *TraCICommandInterface* festgelegt.

```
1 //set speed mode to only regard max acceleration and deceleration
2 traciVehicle->setSpeedMode(6);
```

Listing 6.3: Einstellung des Speed Mode für das CFM.

Durch die Modifikation im Listing 6.3 berechnet das CFM die Geschwindigkeit jedes Automobils nach der Formel

$min(max(SpeedOfStreet) * speedFactor, max(NextSpeed), (maxSpeedOfVehicle))$.

6.3 Algorithmen

In diesem Abschnitt wird die Logik der Implementationen *SimView*, *Update Broadcast*, *Driver* und *CrashVoid* erläutert und der Ablauf im Flussdiagramm dargestellt. Dieses Kapitel bearbeitet die Arbeitspakete 3a und 4a-d, welche im Kapitel 4.5 festgelegt sind. Die Algorithmen, welche durch die Selfmessages implementiert sind, übernehmen die Aufgaben wie unter 5.3.3 beschrieben. Im folgenden sind die Ablaufdiagramme für die Methoden *UpdateData()* 6.1, *AnalyzeData()* 6.2, *getLeader()* 6.3 und *intersectionAnalyze()* 6.4 erläutert. Die vier Methoden sind Bestandteile der Algorithmen *CrashVoid*, *Driver*, *Ausnahmefahrer*, *SimView* und sind an dieser Stelle zentral dargestellt.

Update Broadcast

In diesem Abschnitt wird erläutert wie die Verkehrsteilnehmer periodisch den *Update Broadcast* senden, wodurch das Arbeitspaket 4a bearbeitet wird. Der *Update Broadcast* ist abgeleitet von *WaveShortMessage* (WSM) und *cPacket* (vgl. 2.5.4) und enthält die Daten wie in Tabelle 5.1 einzusehen. Die WSM beinhaltet Informationen, die Veins für die Datenübertragung nutzt, wie Versionsnummer, Chanel Nummer, Datenrate, Priorität und Service ID. Die Variablen aus der Tabelle 5.1 werden wie folgt in der Simulation berechnet oder angefordert; *CarId*, *Geschwindigkeit*, *Koordinaten* und *Direction* stellt das Veins Framework bereit. Die eindeutige Identifikation, die Veins für Simulationsobjekte aus SUMO wählt, wird als *CarId* bezeichnet. Die Fahrzeuglänge ist durch die Veins Erweiterung *getCarLength()* (vgl. 6.2) über das TraCI Interface abrufbar. Die *Geschwindigkeit* und *Position* ist Veins durch die empfangenden Automobile aus SUMO bekannt und die *Beschleunigung* entspricht der Differenz der aktuellen und letzten *Geschwindigkeit* dividiert durch das *Sendeintervall*. Mit jedem *Broadcast* wird die *Position* von Veins angefordert und eine Ungenauigkeit $h, v \in (-gpsImprecision, gpsImprecision)$ zufällig auf die Horizontale und Vertikale *Position* addiert. *TimeStamp* ist mit der Simulationszeit initialisiert, welche mit dem Methodenaufruf *simTime()* von OMNeT++ zu empfangen ist. Mit jedem Broadcast sind die Variablen *Geschwindigkeit, Koordinaten, Ausrichtung, Beschleunigung* und *TimeStamp* neu berechnet oder angefordert. Die Variablen *CarId* und Fahrzeuglänge sind für jedes Automobil innerhalb der Simulation konstant.

Entfernungsberechnung

In diesem Abschnitt wird das Arbeitspaket 4b bearbeitet, das der Anforderung 13 genügen muss (siehe 4.5 und 4.4). In diesem Abschnitt wird das Konzept, welches im

Abschnitt 5.3.5 Unterpunkt *Entfernungsberechnung* ausgearbeitet ist, umgesetzt. Für die Entfernungsberechnung wird in die zwei Methoden *getRealDis2CP()* und *getRealDis()* unterschieden, welche das Kreuzungs- und Auffahrszenario repräsentieren. Für *getRealDis2CP()* wird die aktuelle Broadcast Nachricht des betrachteten Automobil, die Koordinaten von dem ICP und die *Positionsungenauigkeit* benötigt, sodass die Distanz zum ICP durch die Gleichung 6.1 berechnet wird.

$$Distance = fabs(you.getSenderPos().distance(ICP) - PositionImprecision) \quad (6.1)$$

Das Veins Framework stellt die Methode *distance* bereit, mit der die Distanz zwischen zwei Koordinaten berechnet wird. Anschließend wird die Positionsungenauigkeit subtrahiert, wie im Konzept ausgearbeitet. Im Kreuzungsszenario ist der Abstand zum ICP immer positive, wohingegen im Auffahrszenario der Abstand genau dann negative ist, wenn eine Kollision statt findet. Der Abstand zum CP wird durch die Gleichungen 6.2 und 6.3 in der *getRealDis()* berechnet.

$$tmp = you.getSenderPos().distance(other.getSenderPos()) \quad (6.2)$$

$$Distance = tmp - other.getCarLength() - \frac{PositionImprecision}{2} \quad (6.3)$$

Da SUMO die Koordinaten der Verkehrsteilnehmer an der Frontstoßstange angibt muss für das vorausfahrende Automobil *other* die Fahrzeuglänge subtrahiert werden und der Durchschnitt der kombinierten Ungenauigkeit.

Die beiden Methoden sind in dieser Arbeit einheitlich als Metrik dargestellt, sodass je nach Parameter ein kontextabhängiger Aufruf stattfindet. Weiterhin wird in den Ablaufdiagrammen der Positionsungenauigkeitsparameter ausgelassen, wegen dem begrenzten Platz.

Beschleunigungsverhalten

In diesem Abschnitt wird das Arbeitspaket 4c bearbeitet, das der Anforderung 14 unterliegt (siehe 4.5 und 4.4) und das Konzept, welches im Abschnitt 5.3.5 Unterpunkt *Beschleunigungsverhalten* ausgearbeitet ist, umgesetzt. Für das Beschleunigungsverhalten

muss in das Kreuzungs- und Auffahrszenario unterschieden werden, welche durch die Methoden *intersect()*, für das Kreuzungsszenario und *cvBraking()*, für das Auffahrszenario, implementiert ist. In der *intersect()* Methode wird die Zeit zum ICP berechnet, bei aktueller Geschwindigkeit für *you* ($youOldT$) und *other* ($otherOldT$). Anschließend wird die zusätzliche Zeit zum ICP mit der Gleichung 6.4 berechnet, mit der die Kollision vermeidbar ist.

$$safetyTime = \frac{minGab + carLength + carWidth + PositionImprecision}{other.speed} \tag{6.4}$$

Die Variabel *minGab* entspricht dem Gefahrenabstand von 2m, Fahrzeuglänge entspricht *others* Fahrzeuglänge, Breite entspricht 1,5m und die *Positionsungenauigkeit* ist ein variabler Wert (siehe 4.1). Die neue Fahrzeit und Geschwindigkeit wird durch die Gleichung 6.5 und 6.6 berechnet.

$$newT = youOldT + safetyTime - fabs(youOldT - otherOldT) \tag{6.5}$$

$$speedNew = \frac{d(you, ICP, PositionImprecision)}{newT} \tag{6.6}$$

Für die Berechnung von $newT$ wird die Differenz von $youOldT$ und $otherOldT$ subtrahiert, da der betrachtete Fahrer diese Zeit bereits länger zum ICP benötigt, sodass die Bremsung effizienter durchgeführt wird. Dieses Verhalten wird unterbrochen, wenn *safetyTime* kleiner ist als die Differenz von $youOldT$ und $otherOldT$, weil dadurch bestätigt ist, dass mit der aktuellen Geschwindigkeit keine Kollision eintritt. Für diesen Fall wird der Wert *-1* zurückgegeben.

In der *cvBraking()* Methode wird die neue Geschwindigkeit in Abhängigkeit von *Mindestabstand* berechnet (siehe 4.1). Es wird das Verhältnis von aktueller Zeit bis zum CP und *Mindestabstand* gebildet mit den Gleichungen 6.7, 6.8, 6.9.

$$distance = d(you, other, you.imprecision + other.imprecision) \tag{6.7}$$

$$distanceT = \frac{distance}{you.speed} \tag{6.8}$$

$$brakeFactor = \frac{distanceT}{Mindestabstand} \tag{6.9}$$

Nachdem der *brakeFactor* berechnet ist wird untersucht, ob der Abstand größer als der *Mindestabstand* ist, oder kleiner als der Gefahrenabstand. Für ersteres wird der Wert *-1* und für letzteres die Geschwindigkeit 0 zurückgeben. Trifft keine Bedingung zu wird die aktuelle Geschwindigkeit mit *brakeFactor* multipliziert und entspricht der neuen Geschwindigkeit.

UpdateData

In der Abbildung 6.1 ist das Ablaufdiagramm für die Methode *UpdateData()* einzusehen. *UpdateData()* durchläuft die Liste *List* und analysiert die Nachrichten, welche nach der *CarId* des Senders sortiert sind. Durch löschen alter Nachrichten und Nachrichten von Sendern, die zu weit entfernt sind, wird die Liste aktualisiert.

Die Liste *List*, in der Methode *UpdateData()*, wird manipuliert durch das Löschen der Nachrichten, wessen Sender weiter als *dDiff* Meter entfernt ist. In der Abbildung durch die Metrik $d(you, other, you.imprecision + other.imprecision) > dDiff$ dargestellt. Nachrichten, die älter als *tDiff* sind, werden aus der Datenstruktur entfernt. Die Variablen *tDiff* und *dDiff* entsprechen $10 * SendeIntervall$ bzw. 100m.

AnalyzeData

In der Abbildung 6.2 ist die Methode *AnalyzeData()* einzusehen, welche das Ablaufdiagramm für das Arbeitspaket 4d im Abschnitt 4.5 darstellt und der Anforderung 15 genügt. In der *AnalyzeData()* Methode wird die Datenstruktur mit den Broadcast Nachrichten der Verkehrsteilnehmer analysiert und die Automobile extrahiert, mit denen zukünftig interagiert wird (vgl. Konzept Abschnitt 5.3.4).

In der Methode *AnalyzeData()* wird im ersten Schritt untersucht, ob das aktuelle Fahrzeug (*you*) und der Verkehrsteilnehmer (*other*) in die selbe Richtung fahren oder gefahren sind. Dies wird in dem Ablaufdiagramm durch den Operator $\approx$ dargestellt und ist mit einer 10 prozentigen Differenz implementiert. Wenn die Differenz kleiner

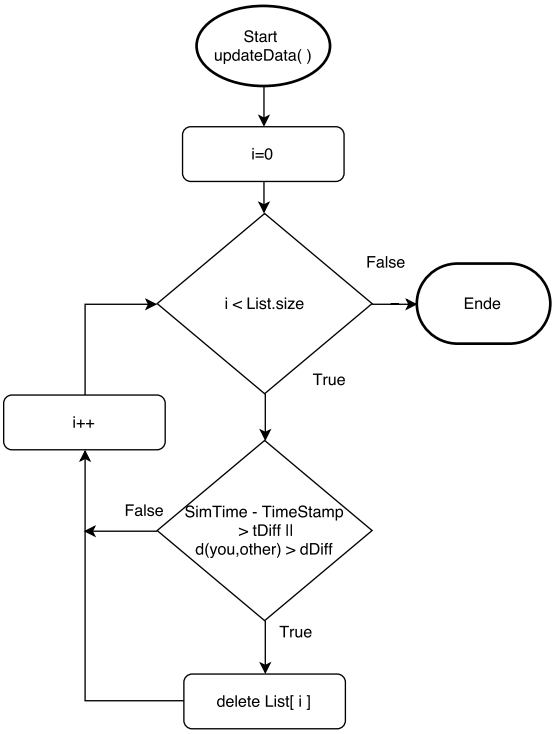


Abbildung 6.1: Ablaufdiagramm für die Methode UpdateData(), welche die Datenstruktur in Abhängigkeit vom Nachrichtenalter und Entfernung zum Sender aktualisiert.

als 10% ist wird ermittelt, ob *other* vor *you* fährt. Fährt *other* vor *you*, wird *other* in der Datenstruktur *Leader* gespeichert, wenn nicht wird die Liste *List* weiter durchlaufen. Ist die Differenz der Fahrrichtung größer als 10% wird ein Kreuzungsszenario mit einem Intersection Collision Point (ICP) auf der Kreuzung angenommen. Im ersten Schritt wird überprüft, ob *you* und *other* einen ICP aufweisen. Haben *you* und *other* einen gemeinsamen ICP und der liegt für beide Automobile voraus, dann wird *other* in die *Leader* Liste gespeichert. Ist der ICP nur für *other* voraus, dann ist *you* bereits auf dem ICP und die Iteration wird fortgesetzt. Ist *other* auf dem ICP, wird durch $d(other, CP, 0) > carWidth + carLength + PositionImprecision$ festgestellt, ob auf eine bevorstehende Kollision zu reagieren ist. Die Metrik $d(other, CP, 0)$ entspricht dem Abstand zwischen *other* und dem ICP. Ist der Abstand größer als die Summe von Fahrzeugbreite, Fahrzeuglänge und *Positionsungenauigkeit*, ist eine Kollision ausgeschlossen.

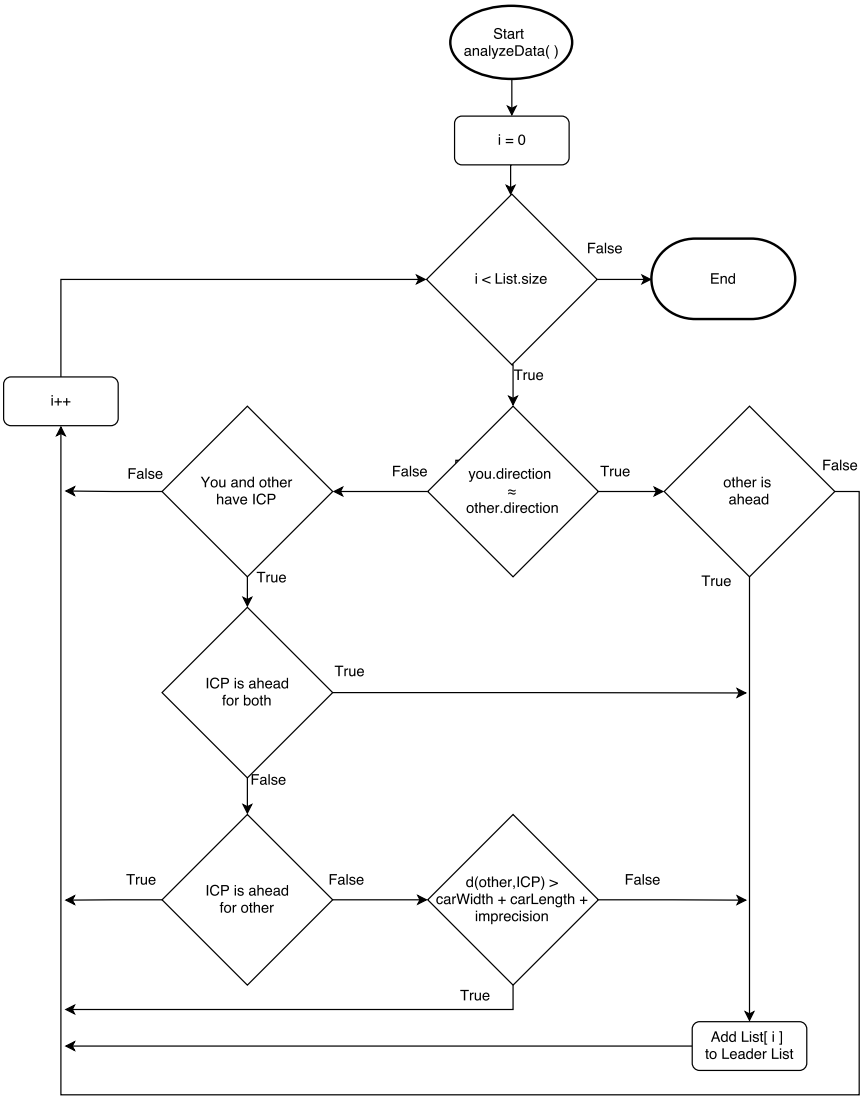


Abbildung 6.2: Ablaufdiagramm der Methode AnalyzeData(), welche aus der Datenstruktur die Automobile extrahiert, mit denen zukünftig interagiert wird.

GetLeader

In der Abbildung 6.3 ist das Ablaufdiagramm der Methode *GetLeader()* einzusehen. Die *GetLeader()* Methode bestimmt aus der *Leader* Liste von *AnalyzeData()* den Verkehrsteilnehmer, für den am stärksten zu bremsen ist.

In der *GetLeader()* Methode wird im ersten Schritt festgestellt, ob die Automobile *you* und *other* einen Intersection Collision Point (ICP) aufweisen oder ein Auffahrunfall

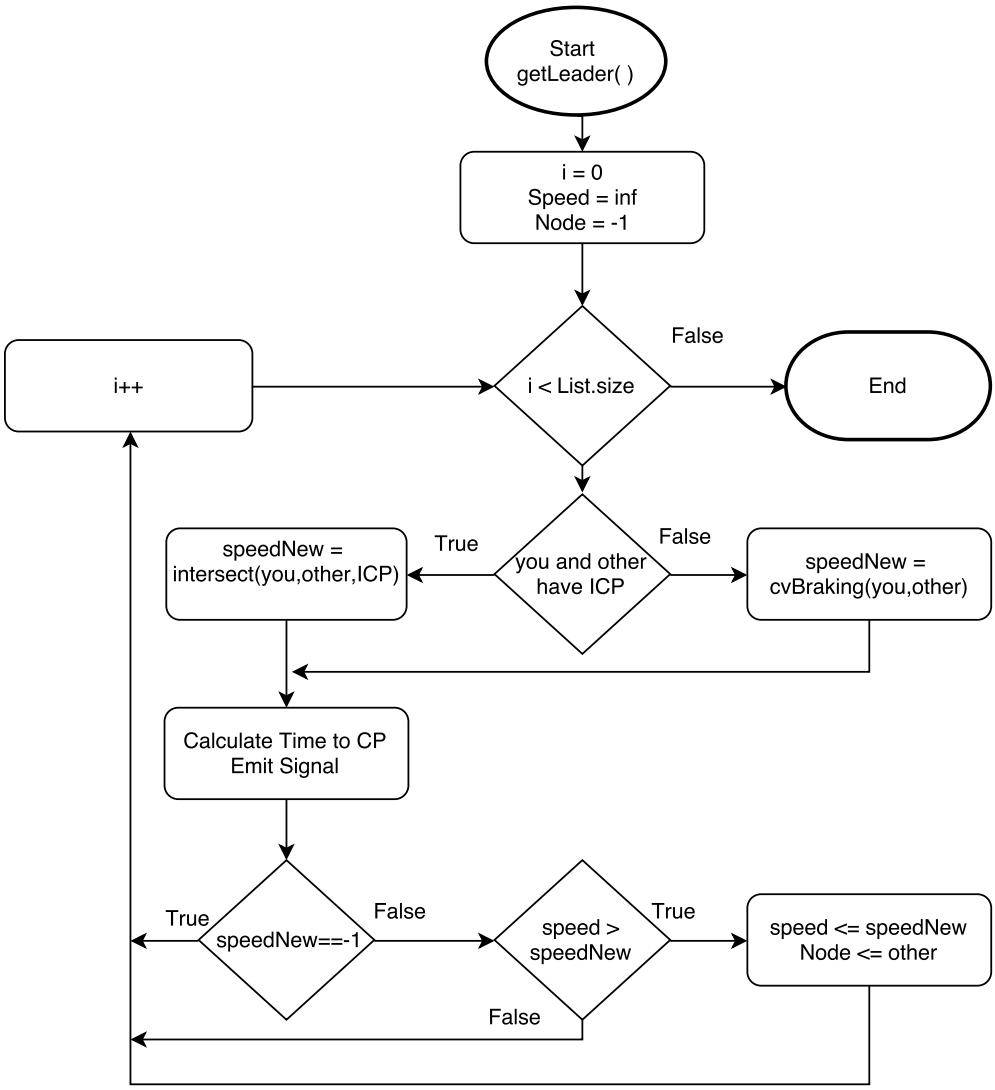


Abbildung 6.3: Ablaufdiagramm der Methode getLeader(), welche den Verkehrsteilnehmer im Szenario bestimmt, für den am stärksten zu bremsen ist.

entstehen kann. Anschließend wird in Abhängigkeit der Situation durch die Methoden *intersect(you, other, ICP)* und *cvBraking(you, other)* die neue Geschwindigkeit berechnet, mit der eine Kollision vermeidbar ist. Die Methoden *Intersect()* und *cvBraking()* signalisieren, wenn keine Kollision bevorsteht, sodass das Durchlaufen der Liste fortgeführt wird. Ist für *other* am stärksten zu bremsen, dann wird *other* lokal gespeichert, indem, wie im Ablaufdiagramm zu erkennen, *Speed* und *Node* neu zugewiesen werden. Nachdem

die Liste durchlaufen ist, ist der Verkehrsteilnehmer festgestellt, für den am stärksten zu bremsen ist, wenn eine Kollision bevorsteht. Abschließend wird die Zeit zum CP (*T2CP*) berechnet. Der Methodenaufruf ist im Listing 6.4 zu sehen.

```
1 //CP on the intersection
2 emit(t2cpInter, getRealDis2CP(you, ICP, 0)/you.getSpeed());
3
4 //CP on rear-end
5 emit(t2cpCVB, getRealDis(you, other, 0)/you.getSpeed());
```

Listing 6.4: Erstellung der Statistiken t2cpInter und t2cpCVB.

Im Listing 6.4 wird durch die Methode *emit()* ein Signal für die statistische Auswertung versendet (vgl. 2.5.5). Der erste Parameter in *emit()* ist der Name der Statistik und der zweite ist der zu dokumentierende Wert, welcher sich aus Abstand zum CP dividiert durch die aktuelle Geschwindigkeit ergibt. Für die Berechnung werden die exakten Koordinaten von *you* und *other* aus dem Simulationsframework benutzt.

IntersectionAnalyze

In der Abbildung 6.4 ist das Ablaufdiagramm der Methode *IntersectionAnalyze()* zu sehen. In der Methode *IntersectionAnalyze()* wird die Kreuzungssituation analysiert und Dead- und Livelocks aufgelöst. Wenn die Verkehrsteilnehmer untereinander im Kreis bremsen, entsteht ein Deadlock, welcher dazu führt, dass Verkehrsteilnehmer an der Kreuzung still stehen. Im ersten Schritt wird durch die Liste *List* iteriert und alle Automobile, die mit *you* einen ICP haben, in die Liste *Copy* gespeichert. Im zweiten Schritt wird die Liste *Copy* durchlaufen und untersucht, ob *Time to Intersection Collision Point* (*T2ICP*) für *you* größer ist als für alle Verkehrsteilnehmer in der Liste *Copy*. Haben die Automobile in der *Leader* Liste ein gleich großes oder größeres *T2ICP* als *you*, dann ist der Live- oder Deadlock durch *you* auflösbar.

6.3.1 *SimView*

SimView fordert in jedem Simulationsschritt Daten der Automobile in der Simulation an. Durch die Daten wie *Geschwindigkeit* und *Koordinaten* werden Statistiken erstellt wie *T2CP*, oder wie viele Kollisionen im Simulationsdurchlauf entstanden sind.

SimView fordern im ersten Schritt eine Liste mit den *Ids* der Simulationsobjekte von SUMO an. Dies wird durch die Methode *getVehicleIds()* implementiert (vgl. 6.2). Im zweiten Schritt wird für jedes Automobil die *Geschwindigkeit*, *Fahrrichtung* und

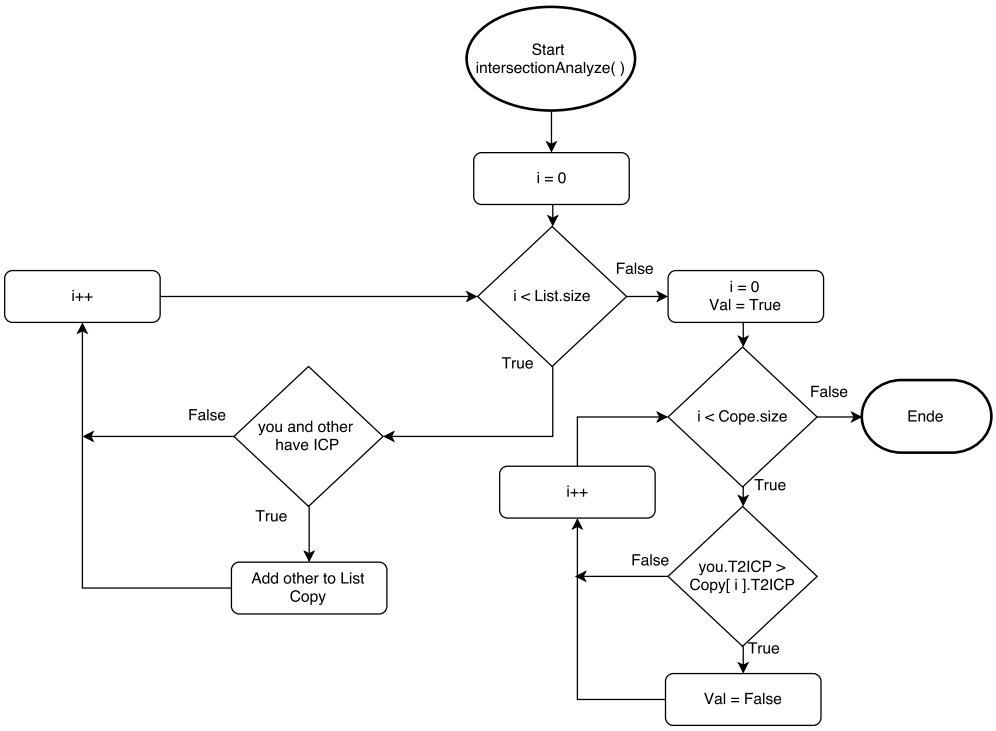


Abbildung 6.4: Das Ablaufdiagramm für die Methode intersectionAnalze(), welche Live- und Deadlocks im Kreuzungsszenario auflöst.

Fahrzeuglänge angefordert und zusammen mit dem *Zeitstempel*, *CarId* und der *Beschleunigung* in die Liste *simViewMap* abgelegt. In der *SimViewMap* sind die Nachrichten für jeden Simulationsschritt nach der *CarId* sortiert. Im dritten und vierten Schritt wird die Methode *UpdateData()* und *AnalyzeData()* aufgerufen, welche im Abschnitt 6.3 im Detail beschrieben sind. Nachdem in der *UpdateData()* Methode die Datenstruktur *SimViewMap* aktualisiert wird und eine Liste *Leader* durch die Methode *AnalyzeData()* erstellt ist, wird im nächsten Schritt festgestellt, ob die Automobile in der Liste *Leader* eine Kollision mit dem aktuellen Automobil haben. Dazu wird in Kreuzungs- und Auffahrszenario unterschieden.

- **Kreuzungsszenario**

  Für eine Kollision im Kreuzungsszenario müssen zwei Bedingungen erfüllt sein. Die erste Bedingung ist für *you* als

  $d(you, ICP, 0) < carWidth$

  definiert und die zweite für *other* als

$d(other, ICP, 0) - other.getCarLength() < carWidth.$

Für die erste Bedingung wird durch die Methode *getRealDis2CP()* berechnet, wie groß der Abstand von *you* zum *ICP* ist, mit einer *Positionsungenauigkeit* von 0m. Alsbald der Abstand kleiner als $carWidth = 1.5m$ ist, ist die aktuelle Situation kollisionsgefährdet. Die zweite Bedingung stellt die Situation dar, wenn *other* den ICP überfahren hat und solange eine Kollision entstehen kann, wie dieser nicht weiter als *carWidth* vom ICP entfernt ist. Die Position der Automobile wird durch SUMO an der Frontstoßstange angegeben, sodass in der Berechnung die *Fahrzeuglänge* subtrahiert wird.

- **Auffahrszenario**

  Für das Auffahrszenario existiert eine Bedingung, welche durch

  $d(you, other, 0) <= 0$

  beschrieben ist. Im Auffahrszenario muss ausschließlich die Distanz zwischen zwei hintereinander fahrenden Automobilen betrachtet werden. Die Methode *getRealDis()* berechnet für das vorausfahrende Fahrzeug *other* und dem folgenden Fahrzeug *you* den Abstand. Alsbald der Abstand zwischen *you* und *other*, bei einer *Positionsungenauigkeit* von 0m kleiner gleich 0m ist, dann ist eine Kollision zwischen den beiden Automobilen eingetreten.

Für beide Szenarien wird im Anschluss die Methode aus dem Listing 6.5 aufgerufen, sodass für die Statistik die Kollision dokumentiert wird.

```
1 emit(Collision, 1.0);
```

Listing 6.5: Erstellung der Statistik Collision.

Im Listing 6.5 wird ein Signal gesendet für die Statistik *Collision* und dem Wert 1, welcher eine Kollision zwischen zwei Automobilen beschreibt.

6.3.2 *CrashVoid*, *Driver* und *Ausnahmefahrer*

Die Algorithmen *CrashVoid*, *Driver* und *Ausnahmefahrer* haben den gleichen Funktionsaufbau und unterscheiden sich in der Interpretation der Berechnungen aus *getLeader()*, sodass an dieser Stelle der Ablauf zentral dargestellt ist.

Die Datenstruktur mit den Broadcast Nachrichten wird von der Methode *UpdateData()* aktualisiert. Anschließend wird durch die Methode *AnalyzeData()* Automobile in die Liste *Leader* extrahiert, welche eine Wahrscheinlichkeit haben, mit dem aktuellen Fahrzeug eine

Kollision zu haben. In der *getLeader()* Methode wird aus der *Leader* Liste das Fahrzeug extrahiert, für das am stärksten zu bremsen ist. Die *IntersectionAnalyze()* Methode löst Dead- und Livelocks auf, die an der Kreuzung stattfinden, wenn die *getLeader()* Methode kein Ergebnis liefert. Hat die *IntersectionAnalyze()* Methode einen Live- oder Deadlock erkannt, wird dem aktuellen Automobil der Befehl gegeben, wie im CFM definiert, zu fahren und der Algorithmus wird beendet. Ist der Live- oder Deadlock nicht durch das aktuelle Fahrzeug auflösbar, wird der Algorithmus beendet, ohne das Fahrverhalten zu manipulieren.

Driver

Der *Driver* Algorithmus bewegt das Automobil durch das Szenario, basierend auf den Koordinaten, die durch den *Simview* Algorithmus erstellt werden. In Abhängigkeit der Fahrparameter *Geschwindigkeit* und *Beschleunigung* des Verkehrsteilnehmer, der durch *getLeader()* berechnet ist, wird durch *Driver* die Zuständigkeit festgestellt. Ist die Bedingung $Geschwindigkeit = 0$ && $Beschleunigung < 0$ falsch, dann führt *Driver* die Geschwindigkeitsänderung durch.

CrashVoid

Der CrashVoid Algorithmus analysiert die empfangenen Broadcast Nachrichten und manipuliert das Verhalten des Automobils, wenn eine Kollision bevorsteht. Das Verhalten des *CrashVoid* Algorithmus ist durch die Fahrparameter *Geschwindigkeit* und *Beschleunigung* des Automobils aus der *getLeader()* Methode beeinflusst. Der *CrashVoid* Algorithmus ignoriert das Automobil, wenn die $Geschwindigkeit > 0$ && $Beschleunigung \geq 0$ wahr ist. Trifft die Bedingung für das Automobil nicht zu, wird es in die Liste *reaction* gespeichert. Ist das enthaltene Automobil in *reaction* älter als *Reaktionszeit* Sekunden (vgl. 4.1), dann wird die neu berechnete Geschwindigkeit angewendet.

Ausnahmefahrer

Ausnahmefahrer implementieren den *CrashVoid* Algorithmus und setzen die Ergebnisse ausschließlich um, wenn das betreffende Automobil im Auffahrszenario ist.

7 Evaluierung

Dieses Kapitel stellt die Simulationsergebnisse vor, die durch die Simulation des Assistenzsystems entstanden sind. Im Kapitel Analyse 4 ist im Abschnitt 4.5 das Arbeitspaket 4 definiert, das festlegt, welche Variablen im Assistenzsystem zu dokumentieren sind. Das Kapitel Implementierung 6 zeigt, wie das Assistenzsystem implementiert ist und wie die statistischen Variablen erfasst werden, sodass in diesem Kapitel die Auswertung der Dokumentation stattfindet.

Dieses Kapitel unterteilt sich in die vier Abschnitte *Durchführung*, *Szenario*, *Übertragungsmedium* und *Leistungsfähigkeit des Assistenzsystems*. Der Abschnitt *Durchführung* 7.1 fasst zusammen, mit welchem Computer, Betriebssystem und Software Versionsnummern die Simulation durchgeführt wird, sodass die Simulationsergebnisse reproduzierbar sind. Der Abschnitt *Szenario* wertet die Zeit und gefahrene Strecke von den Verkehrsteilnehmer aus. Der Abschnitt Übertragungsmedium 7.3 stellt die Statistiken für das Übertragungsmedium in Diagrammen dar und bewertet die Ergebnisse bezüglich der effizienten Nutzung. Im Abschnitt *Leistungsfähig des Assistenzsystems* 7.4 sind die Statistiken für die Kollisionsfrüherkennung und der entstandenen Kollisionen dargestellt mit abschließender Bewertung.

7.1 Durchführung

Die Simulation ist durchgeführt mit einem HP Elite Book 8740w, 8Gb RAM und Intel Core I7 CPU M 640 mit einem 2,8GHz Dual Core. Als Betriebssystem wird ein 64Bit Ubuntu 14.04 LTS verwendet. Für die Durchführung der Simulation wird OMNeT++ 5.0 Build Id 160414-aa4629c, Veins 4.4 und SUMO 0.25.0 verwendet.

Weiterhin wird die Simulation mit den 802.11.p Physical Layer Einstellungen aus dem Listing 7.1 durchgeführt.

```
1 *.**.nic.mac1609_4.txPower = 20mW
2 *.**.nic.mac1609_4.bitrate = 18Mbps
3 *.**.nic.phy80211p.thermalNoise = -110dBm
4 *.connectionManager.alpha = 2.0
```

```
5 *.connectionManager.carrierFrequency = 5.890e9 Hz
```

Listing 7.1: Sendereinstellung für 802.11p Physical Layer im OMNeT++ initialization File.

Das Listing 7.1 zeigt die 802.11.p Physical Layer Einstellungen, wobei *txPower* für Transmitter Power Control steht und die Sendeleistung definiert. Die *bitrate* beschreibt den Datendurchsatz in Megabit pro Sekunde. *thermalNoise* beschreibt das Hintergrundrauschen im Szenario und *alpha* den minimalen Pfadverlust. *carrierFrequency* legt die Frequenz fest, in der gesendet wird.

7.2 Szenario

Dieser Abschnitt stellt die Statistiken *Total Time* (TT) und *Total Distance* (TD) der Verkehrsteilnehmer vor. TT stellt die durchschnittliche Zeit in Sekunden pro Verkehrsteilnehmer im Szenario und Simulationsdurchlauf dar und TD ist die zugehörige durchschnittlich gefahrene Distanz in Metern. Die Tabelle 7.1 zeigt die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall der Statistiken TT und TD.

Name	**Stichprobengröße**	**Durchschnitt**	**Std. Abweichung**	**Konfidenzintervall**
TT	110182	38,36	15,95	0,094
TD	110182	341,35	96,4	0,569

Tabelle 7.1: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistiken TT und TD.

.

Tabelle 7.1 zeigt für die Statistiken TT und TD eine Stichprobengröße von 100.000 und ein Konfidenzintervall von 0,0025% und 0,0006% relativ zu dem arithmetischen Mittel von TT bzw. TD. Weiterhin ist zu erkennen, dass die Größe der Standardabweichung mit 41,58% und 28,24% relativ zum arithmetischen Mittel für TT bzw. TD klein ist, sodass die Stichproben für beide Statistiken konzentriert liegen. Die Tabelle 7.1 zeigt, dass TD eine 12,6% kleinere durchschnittliche Strecke aufweist, als die im Abschnitt 4.6.4 ausgearbeiteten 390,54m. Dieses Verhalten ist auf die Kollisionen im Auffahrszenario zurückzuführen, wie im Kapitel 4.3.2 Abschnitt *Physikalisches Verhalten in SUMO* erläutert. Aus dieser Argumentation folgt, dass jedes Fahrzeug durchschnittlich um 49,19m nach vorne verschoben wird durch SUMO. Außerdem zeigt die Tabelle 7.1, dass der Erwartungswert für die Zeit im Szenario bei 38,36 Sekunden liegt und für

68,2% (eine Standardabweichung) der Verkehrsteilnehmer zwischen 22,41 und 54,31 Sekunden ist, sodass mit der gefahrenen Strecke von 244,95 und 437,75 Metern (eine Standardabweichung) eine Geschwindigkeit zwischen 10,93$\frac{m}{s}$ und 8,06$\frac{m}{s}$ erreicht wird.

7.2.1 Bewertung

Es ist zu erkennen, dass die Geschwindigkeiten der Verkehrsteilnehmer verteilt sind, sodass das Szenario unterschiedliche Stadtszenarien erfolgreich simuliert, durch die Verfügbarkeit von Abschnitten mit hohen und niedrigen Geschwindigkeiten, was auch durch das Einfügen der Verkehrsteilnehmer gesteuert ist, wie im Abschnitt 4.6.3 dargelegt.

7.3 Übertragungsmedium

Dieses Kapitel stellt die Simulationsergebnisse für die Auslastung des Übertragungsmediums dar. Das Verhalten des Mediums statistisch zu dokumentieren ist als Arbeitspaket 4 4.5 im Kapitel 4 festgelegt. Das Verhalten ist durch Veins in Form von *Times Into Backoff* (TIB), *Received Broadcasts* (RBC), gesendete Broadcasts (*Total Send* TS), Anzahl der gewarteten Slots (*Waited Slots* WS) und *Total Lost Packets* (TLP) dokumentiert. In diesem Kapitel werden die Simulationsergebnisse für das Übertragungsmedium graphisch dargestellt, analysiert, interpretiert, sodass abschließend die Simulationsergebnisse bewertet werden, um die Hypothesen aus dem Kapitel 4.7 zu belegen.

7.3.1 Statistiken *TLP* und *RBC*

Das Diagramm 7.1 zeigt die empfangenen *Broadcast Nachrichten* (RBC) und *Total Lost Packets* (TLP), welche einen Bit-Error haben oder während des Sendevorgangs empfangen sind, in Abhängigkeit von dem *Senderintervall*. TLP und RBC entsprechen der durchschnittlichen Summe pro Verkehrsteilnehmer und Simulation an empfangenen und fehlerbehafteten Broadcasts. TLP und RBC weisen eine Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall wie in der Tabelle 7.2 auf.

Der Durchschnitt (Erwartungswert) von TLP und RBC, aus der Tabelle 7.2, ist im Diagramm 7.1 dargestellt. Die Statistiken für 1Hz und 2Hz und RBC 10Hz zeigen eine Standardabweichung in der Größenordnung vom Durchschnitt woraus gefolgert wird, dass die Stichproben konzentriert um den nach links verschobenen Erwartungswert liegen. Die Standardabweichung für TLP ist um den Faktor 1,47 größer als der Durchschnitt, sodass die Verteilung weiter nach links verschoben ist und die Stichproben, die größer

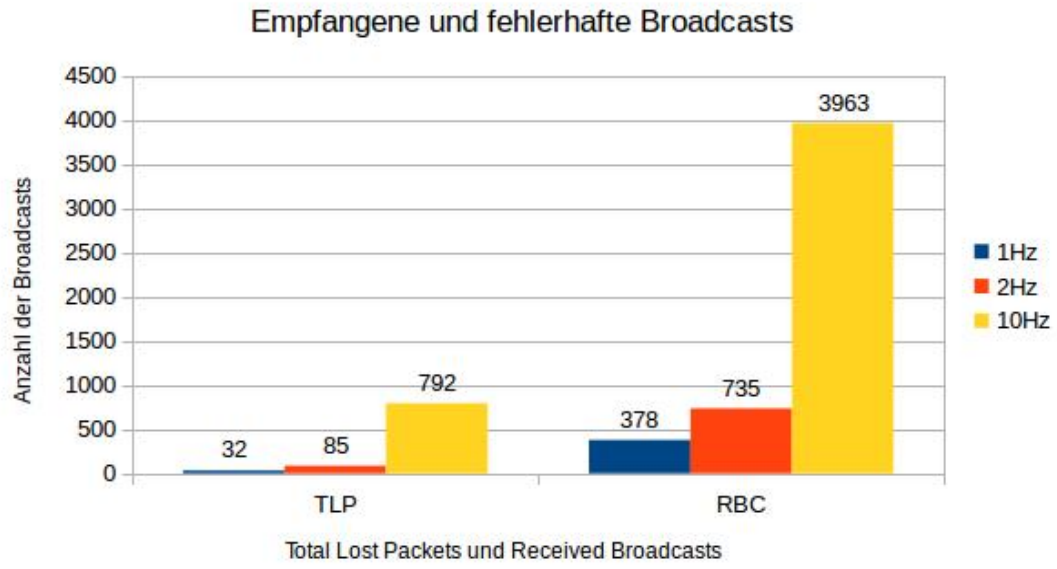


Abbildung 7.1: Das Diagramm zeigt das Verhalten des Übertragungsmedium im Bezug auf empfangene Broadcast Nachrichten und den Anteil der Fehlerbehaftet ist in Abhängigkeit vom Sendeintervall.

Name	**Stichprobengröße**	**Durchschnitt**	**Std. Abweichung**	**Konfidenzintervall**
TLP 1Hz	37058	32	55,9	0,569
TLP 2Hz	36852	85	89	0,91
TLP 10Hz	36272	792	1165,7	11,99
RBC 1Hz	37058	378	275,7	2,81
RBC 2Hz	36852	735	447,1	4,57
RBC 10Hz	36272	3963	3517,8	36,2

Tabelle 7.2: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistiken RBC und TLP in dem Diagramm 7.1.

.

als der Erwartungswert, sind verteilter liegen. Das Konfidenzintervall liegt für die sechs Statistiken in der Größenordnung von 1%, sodass die Stichprobengröße ausreichend ist um aus dem Diagramm statistisch Zusammenhänge abzuleiten.

Das Diagramm zeigt, dass RBC, wie das Sendeintervall, ca. mit den Faktoren 2 und 5 auf 378, 735 und 3963 empfangener Broadcasts wächst. TLP ist mit der Verdoppelung des *Senderintervalls* um den Faktor 2,65 von 32 auf 85 und mit der Verfünffachung um den Faktor 9,32 auf 792 fehlerhafte Nachrichten gestiegen, sodass für 1Hz, 2Hz und 10Hz 8,5%, 11,56% bzw. 20% aller Broadcasts fehlerhaft sind.

7.3.2 Statistiken *TIB*, *WS* und *TS*

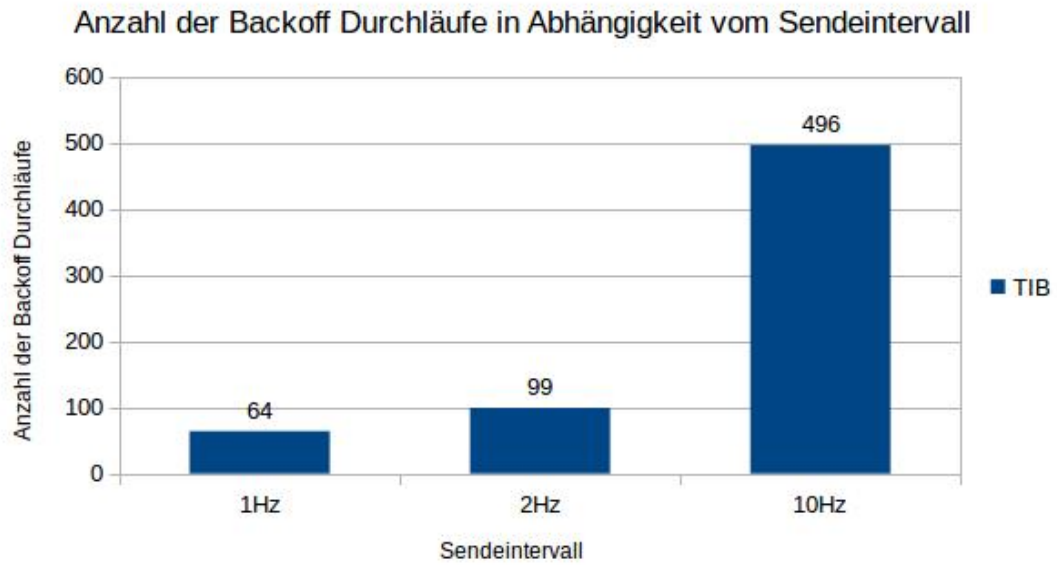


Abbildung 7.2: Das Diagramm zeigt die durchschnittliche Anzahl an Backoff Durchläufen in Abhängigkeit vom Sendeintervall.

Das Diagramm 7.2 zeigt die Anzahl der Backoff-Durchläufe (TIB) in Abhängigkeit vom Sendeintervall (vgl. 2.2.2). TIB stellt die durchschnittliche Summe pro Verkehrsteilnehmer und Simulation an Backoff-Durchläufen dar. Die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall von TIB ist in der Tabelle 7.3 einzusehen.

Die Tabelle 7.3 zeigt, dass die Standardabweichung für die drei TIB Statistiken kleiner als bis halb so groß wie der Erwartungswert ist, sodass die Verteilung für TIB 1Hz nach

Name	Stichprobengröße	Durchschnitt	Std. Abweichung	Konfidenzintervall
TIB 1Hz	37058	64	61,6	0,63
TIB 2Hz	36852	99	58,3	0,6
TIB 10Hz	36272	496	282,2	2,9

Tabelle 7.3: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik TIB in dem Diagramm 7.2.

links verschoben ist und für 2Hz und 10Hz die Verteilung der Stichproben dichter um den Erwartungswert sind. Das Konfidenzintervall liegt für die drei Statistiken unter 1% relativ zum Durchschnitt, sodass die Stichprobengröße ausreichend groß ist, um das Verhalten von TIB statistisch zu deuten.

Das Diagramm 7.2 zeigt den Verlauf von TIB, der vergleichbar mit dem *Sendeintervall* ansteigt. Mit der Verdoppelung des *Sendeintervalls* von 1Hz auf 2Hz steigt TIB um 154,7% an und mit der Verfünffachung auf 10% um 501%.

Das Diagramm 7.3 zeigt die durchschnittliche Summe der gewarteten Slots im Backoff (Waited Slots WS) Algorithmus pro Verkehrsteilnehmer und Simulationsdurchlauf in Abhängigkeit vom *Sendeintervall*. In Tabelle 7.4 ist die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall der Statistiken aus dem Diagramm 7.3 einsehbar.

Name	Stichprobengröße	Durchschnitt	Std. Abweichung	Konfidenzintervall
WS 1Hz	36272	222,46	216,49	2,20
WS 2Hz	36852	345,05	205,27	2,96
WS 10Hz	23001	1734,35	989,18	10,18

Tabelle 7.4: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik WS in Abhängigkeit vom Sendeintervall in dem Diagramm 7.3.

In der Tabelle 7.4 ist zu erkennen, dass die Verteilung für WS 1Hz mit einer Standardabweichung, die eine vergleichbare Größe wie der Erwartungswert aufweist, nach links verschoben ist, sodass die Stichproben kleiner als der Durchschnitt konzentriert liegen und größer als der Durchschnitt nach rechts verteilen. Für WS 2Hz und 10Hz ist

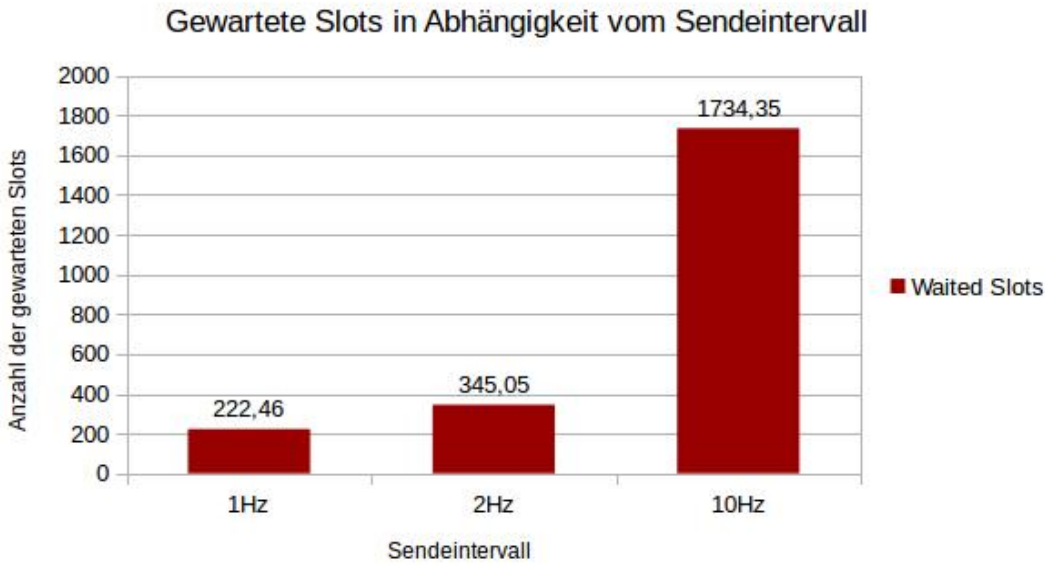


Abbildung 7.3: Das Diagramm zeigt die durchschnittliche Summe an gewarteten Slots pro Verkehrsteilnehmer und Simulationsdurchlauf in Abhängigkeit vom Sendeintervall.

die Standardabweichung relativ kleiner und somit sind die Stichproben breiter um den Erwartungswert verteilt. Das Konfidenzintervall ist mit unter einen Prozent für die drei Statistiken klein, welches auf die Anzahl der Stichproben zurückzuführen ist.

Das Diagramm 7.3 zeigt für 1Hz 222,46 Slots und steigt um 55,1% mit der Verdoppelung der Frequenz auf 2Hz. Mit der Verfünffachung des *Sendeintervalls* auf 10Hz nimmt die Anzahl der gewarteten Slots um 502,6% zu und entspricht ebenfalls der Verfünffachung. Es ist zu erkennen, dass die Statistiken WS und TIB (vgl. Diagramm 7.2) ein ähnliches Verhalten zeigen, woraus geschlossen wird -mit Hilfe des Diagramms 7.1- dass der Zugriff auf das Medium ineffizient ist, weil die gewarteten Slots pro Backoff Durchlauf mit 3,48 (1Hz), 3,49 (2Hz) und 3,5 (10Hz) ähnlich sind, obwohl, wie im Diagramm 7.2 zu erkennen, die empfangenen Broadcasts mit 1Hz und 10Hz Sendefrequenz um den Faktor 10,5 verschieden sind.

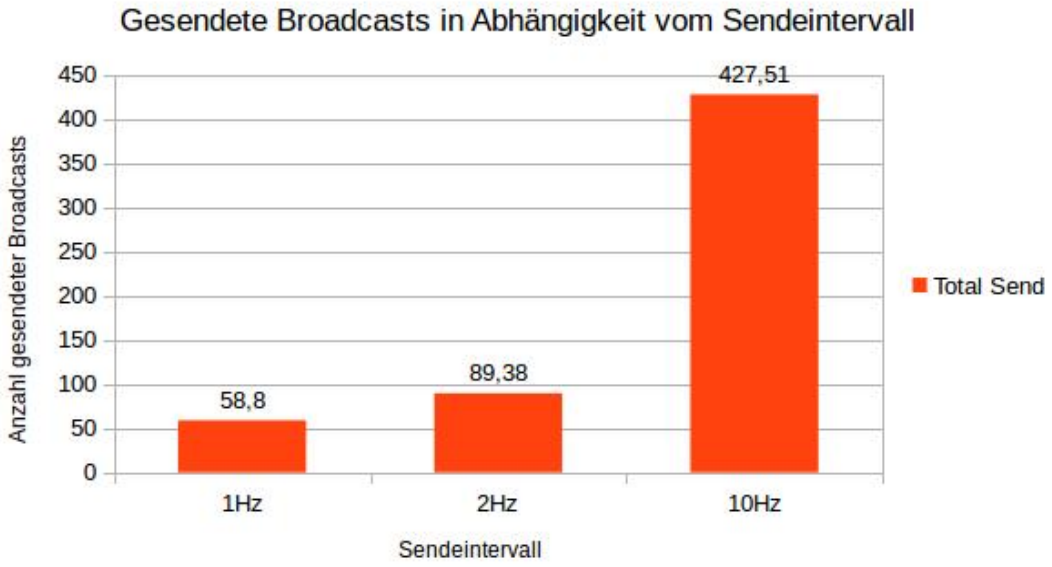


Abbildung 7.4: Das Diagramm zeigt die durchschnittliche Summe an gesendeten Broadcasts pro Verkehrsteilnehmer und Simulationsdurchlauf in Abhängigkeit vom Sendeintervall.

Das Diagramm 7.4 zeigt die durchschnittliche Summe der gesendeten Broadcasts *Total Send* (TS) pro Verkehrsteilnehmer und Simulationsdurchlauf in Abhängigkeit vom *Sendeintervall*. In Tabelle 7.5 ist die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall der Statistiken aus dem Diagramm 7.4 einsehbar.

Name	Stichprobengröße	Durchschnitt	Std. Abweichung	Konfidenzintervall
TS 1Hz	37058	58,80	55,78	0,57
TS 2Hz	36852	89,38	53,09	0,54
TS 10Hz	36272	427,51	209,19	2,15

Tabelle 7.5: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik TS in Abhängigkeit vom Sendeintervall in dem Diagramm 7.4.

Die Tabelle 7.5 zeigt, dass die Statistik TS 1Hz aufgrund der relativ großen Standardabweichung zum Durchschnitt nach links verschoben ist, sodass sich die Stichproben, die kleiner als der Erwartungswert sind, konzentriert liegen. Für die Statistiken TS 2Hz und 10Hz ist die Standardabweichung kleiner relativ zum Durchschnitt, sodass die Stichproben gleichmäßiger verteilt sind. Die Stichprobengröße ist in der Größenordnung von über 36.000, sodass das Konfidenzintervall mit unter 1% für die drei Statistiken klein ist.

Das Diagramm 7.4 zeigt die gesendeten Broadcast von 58,8, 89,38 und 427,51 für die *Sendefrequenzen* von 1Hz, 2Hz und 10Hz. Mit der Verdoppelung von 1Hz auf 2Hz *Sendeintervall* werden 52% mehr Broadcasts versendet und mit der Verfünffachung von 2Hz auf 10Hz 478,78%, welches ein geringeres Wachstum als das *Sendeintervall* entspricht, woraus geschlossen wird, dass für 2Hz 48% und für 10Hz 21,22% der Nachrichten länger als 16 Durchläufe im Backoff Algorithmus gewartet haben und anschließend nicht mehr gesendet worden (vgl. 2.2.2). Dies entspricht für 10Hz 115,15 und 2Hz 82,5 Nachrichten und ist mit einen Unterschied von 39,6% in der gleichen Größenordnung. Mit dem Erwartungswert der drei Statistiken ergibt sich aus dem Diagramm 7.2 die Anzahl der Backoff-Durchläufe pro Broadcast für 1Hz, 2Hz und 10Hz als 1,09, 1,11 und 1,16. Mit den Berechnungen der durchschnittlich gewarteten Slots pro Backoff Durchlauf, in Abhängigkeit vom *Sendeintervall* 1Hz, 2Hz und 10Hz, von 3,48, 3,49, und 3,5, für das Diagramm 7.3, ergibt sich eine durchschnittliche Wartezeit pro Broadcast von 3,79, 3,87 und 4,06 Slots.

7.3.3 Bewertung

Das Diagramm 7.4 hat gezeigt, dass aufgrund eines belegten Übertragungsmedium für ein *Sendeintervall* von 2Hz und 10Hz 48% bzw. 21,22% der Nachrichten verworfen werden, weil die Wartezeit zu groß ist. Aus den Diagrammen 7.4, 7.3 und 7.2 wird

gefolgert, dass jede Broadcast-Nachricht, die gesendet wird, in Abhängigkeit vom *Sendeintervall* durchschnittlich 3,79, 3,87 oder 4,06 Slots wartet, bevor die Daten auf das Übertragungsmedium modelliert werden. Das Diagramm 7.1 zeigt, dass der Anteil der empfangenen Broadcast-Nachrichten abhängig vom *Sendeintervall* zu 8,5%, 11,56% bzw. 20% fehlerhaft sind.

Aus den Forschungsergebnissen ist zu erkennen, dass der verwendete CSMA/CA MAC Algorithmus keinen effizienten Zugriff auf das Medium bereit stellt, weil im Backoff-Algorithmus 48% bzw. 21,22% der Nachrichten verworfen werden. Gesendete Nachrichten haben eine durchschnittliche Verspätung von 3,79-4,06 Slots. Von den 52% und 78,78% der empfangenen Nachrichten haben 11,56% bzw. 20% einen Bit-Error, sodass für ein 2Hz *Sendeintervall* 45,99% und für das 10Hz *Sendeintervall* 63,01% der Nachrichten fehlerfrei zugestellt werden. Durch diese Ausarbeitung sind die Eigenschaften des CSMA/CA Algorithmus weiter erforscht und es hat sich gezeigt, dass der Channel-Zugriff, welcher durch AIFS und Backoff Algorithmus implementiert ist, fehleranfällig ist und, wie in [52] dargestellt, ungeeignet für Safety-Anwendung im C2X Kontext ist, aufgrund der fehlenden Worst-Case Kanalzugriffszeit. Die fehlende Worst-Case-Zugriffszeit wird durch diese Ausarbeitung weiterhin bestätigt[1] und es wird weiterhin gezeigt, dass der CSMA/CA Algorithmus für ein gering ausgelastetes Medium stark fehleranfällig ist und die Fehleranfälligkeit für ein ausgelastetes Medium relativ gering ansteigt[2]. Es ist zu erkennen, dass die Leistungsfähigkeit des Assistenzsystems abhängig von der effizienten Ausnutzung des Übertragungsmediums ist, sodass die entscheidenden Defizite im CSMA/CA Algorithmus die Bandbreiten Zuordnung durch AIFS und Backoff Algorithmus sind, die keinen Algorithmus für einen deterministischen Channel Zugriff bereit stellen, sodass gezeigt ist, dass Anforderungen 10 und 11 nicht erfüllt sind.

7.4 Leistungsfähigkeit des Assistenzsystems

In diesem Abschnitt wird die Leistungsfähigkeit des Assitenzsystem untersucht bezüglich der Früherkennung von Kollisionen und den entstandenen Kollisionen. Die Zeit bis zur Kollision (*T2CP*) und die entstandenen Kollisionen (*Kollisionen*) zu dokumentieren, ist in dem Arbeitspaket 4 4.5 im Kapitel 4 festgelegt. Die Statistiken sind wie im Kapitel 6 erstellt und in diesem Abschnitt graphisch dargestellt, analysiert und interpretiert, um

[1] Im Backoff Algorithmus werden zwischen 21,22% und 48% der Nachrichten verworfen (2Hz bzw. 10Hz)
[2] Für ein 2Hz und 10Hz Sendeintervall sind 45,99% bzw. 63,01% der Nachrichten fehlerfrei empfangen

durch die Simulationsergebnisse die Hypothesen aus dem Kapitel 4.7 zu belegen und daraufhin die Leistungsfähigkeit des Assistenzsystems einzuordnen.

7.4.1 Statistik *T2CP*

Dieser Abschnitt stellt die Simulationsergebnisse für die Statistik *T2CP* dar, welche die Zeit ab dem Erkennen bis zum Eintreten einer Kollision beschreibt. Für alle Statistiken von *T2CP* sind die Simulationsergebnisse gefiltert. Die folgenden Diagramme verwenden die Simulationsergebnisse von der Statistik *T2CP*, bei denen alle Ergebnisse über 10s entfernt sind, weil durch Ausrollen an der Kreuzung, oder dem langsamen Auffahren zum vorausfahrenden Verkehrsteilnehmer die Statistik positives verfälscht wird. Im diesem Abschnitt wird die Statistik *T2CP* in Abhängigkeit der Iterationsvariablen *Sendeintervall*, *Mindestabstand* und *Positionsungenauigkeit* in Diagrammform dargestellt.

Iterationsvariable *Sendeintervall*

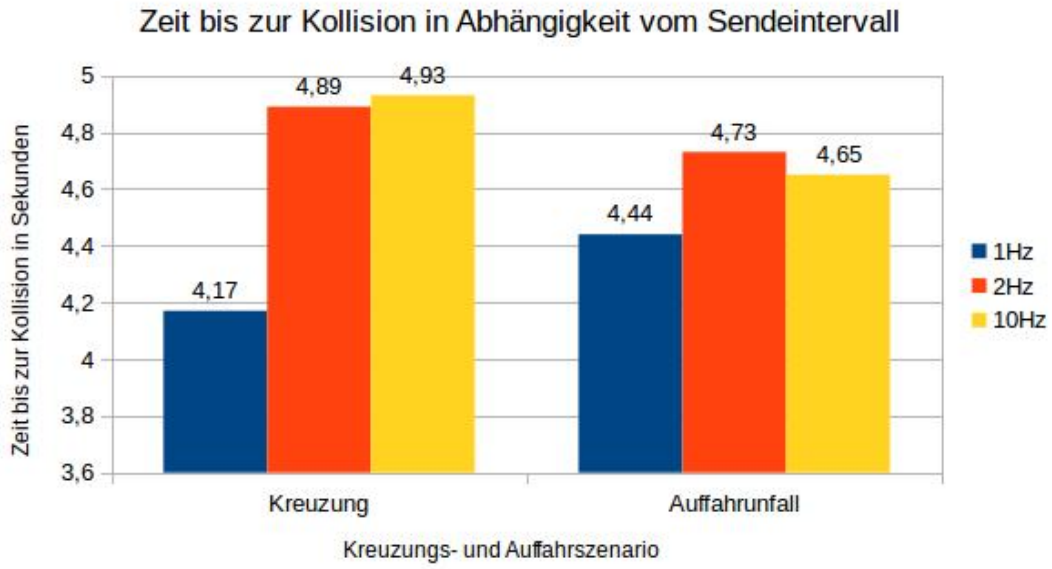


Abbildung 7.5: Das Diagramm zeigt die Zeit bis zum eintreten der Kollision in Abhängigkeit vom Sendeintervall.

Das Diagramm 7.5 zeigt die Zeit bis zum Eintreten einer Kollision in Abhängigkeit vom *Sendeintervall* für das Kreuzungs- und Auffahrszenario. In der Tabelle 7.6 ist die

Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für das Diagramm 7.5 einzusehen.

Name	**Stichprobengröße**	**Durchschnitt**	**Std. Abweichung**	**Konfidenzintervall**
Kreuzung 1Hz	21975	4,17	2,25	0,03
Kreuzung 2Hz	24098	4,89	2,14	0,03
Kreuzung 10Hz	23001	4,92	1,91	0,02
Heck 1Hz	20831	4,44	1,87	0,03
Heck 2Hz	22186	4,78	1,97	0,03
Heck 10Hz	19749	4,65	7,64	0,11

Tabelle 7.6: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik T2CP in Abhängigkeit vom Sendeintervall in dem Diagramm 7.5.

Die Tabelle 7.6 zeigt, dass die Standardabweichungen für alle Statistiken ca. um den Faktor 0,5 kleiner und für das Auffahrszenario mit 10Hz um 1,64 größer ist als der Durchschnitt. Innerhalb einer Standardabweichung sind 68,2% aller Stichproben, somit sind die Stichproben für Heck 10Hz nach links verschoben, konzentriert um den Erwartungswert und die Stichproben größer als der Durchschnitt sind verteilter. Das Konfidenzintervall liegt für alle Statistiken unter 1% und für 10Hz Heck bei 2,4%, sodass die Stichprobengröße ausreichend groß ist und aus dem Diagramm statistische Aussagen ableitbar sind.

Das Diagramm 7.5 zeigt für das Kreuzungsszenario, dass mit der Verdoppelung des *Sendeintervalls* von 1Hz auf 2Hz eine Kollision 0,72 Sekunden und mit einer Verfünffachung von 2Hz auf 10Hz nur noch 0,04 Sekunden früher erkannt wird. Für das Auffahrszenario steigt die Früherkennung um 0,29 Sekunden, wenn das *Intervall* von 1Hz auf 2Hz verdoppelt wird und nimmt um 0,08 Sekunden ab, wenn das *Senderintervall* von 2Hz auf 10Hz verfünffacht wird. Die Leistungsminderung ist auf das stärker ausgelastete Medium zurückzuführen, dass durch das Ansteigen des Faktors $\frac{WS}{TS}$ beschrieben wird (siehe Diagramme 7.1, 7.2 und 7.3). In dem Diagramm 7.5 ist zu erkennen, dass der Leistungszuwachs zwischen 1Hz und 2Hz relativ größer ist, als der Zuwachs zwischen 2Hz

und 10Hz. Aus dem Diagramm 7.1 ist zu erkennen, dass mit steigendem *Sendeintervall* die Anzahl der Nachrichten mit Bit-Error auf 8,5%, 11,56% und 20% ansteigt, sodass durch den ineffizienten Channel-Zugriff und Ausnutzung der verfügbaren Bandbreite das Potenzial des *Sendeintervalls* unzureichend genutzt wird. Aus der Gleichung 4.1 im Abschnitt 4.6.4 wird gefolgert, dass Bremsvorgänge von Automobilen erkannt werden, die durchschnittlich 3,39 Automobile vorausfahren, 3,98 und 4,01 für die *Sendeintervalle* 1Hz, 2Hz bzw. 10Hz im Kreuzungsszenario. Für das Auffahrszenario ergibt sich eine Früherkennung von 3,61, 3,85 und 3,78 Automobilen für 1Hz, 2Hz und 10Hz, sodass gezeigt ist, dass die Anforderung 16 im Abschnitt 4.4 eingehalten ist.

Iterationsvariable *Mindestabstand*

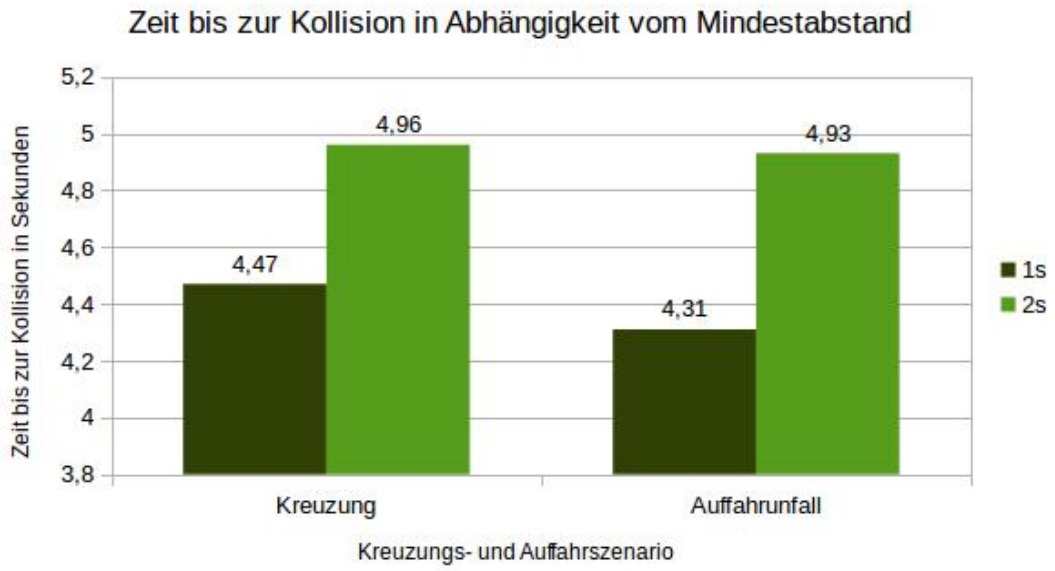


Abbildung 7.6: Das Diagramm zeigt die Zeit bis zum eintreten einer Kollision in Abhängigkeit von dem Mindestabstand an.

Das Diagramm 7.6 zeigt die Entwicklung von T2CP in Abhängigkeit vom *Mindestabstand* im Kreuzungs- und Auffahrszenario. In der Tabelle 7.7 ist die Stichprobenmenge, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik Mindestabstand einzusehen.

Name	Stichprobengröße	Durchschnitt	Std. Abweichung	Konfidenzintervall
Kreuzung 1s	40000	4,47	1,87	0,02
Kreuzung 2s	29074	4,96	2,41	0,03
Heck 1s	30791	4,31	1,9	0,02
Heck 2s	31975	4,93	6,12	0,08

Tabelle 7.7: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik T2CP in Abhängigkeit vom Mindestabstand in dem Diagramm 7.6.

Aus der Tabelle 7.7 ist zu entnehmen, dass die Stichproben der Statistiken um den Erwartungswert verteilt sind mit einer Standardabweichung die ca. halb so groß ist wie der Durchschnitt. Für das Auffahrszenario und 2s *Mindestabstand* ist eine Standardabweichung von 6,12 gegeben, bei einem Durchschnitt von 4,93. Zu erkennen ist, dass die Stichproben die kleiner als der Erwartungswert sind, dicht am Durchschnitt liegen. Stichproben größer als der Erwartungswert sind verstreuter. Dies wird weiterhin vom größeren Konfidenzintervall 0,08 bestätigt, welches mit 1,6%, relativ zum Durchschnitt, größer ist als das Konfidenzintervall von den Kreuzungsszenarien mit 0,4% und 0,6% für 1s bzw. 2s und das Auffahrszenario mit 0,4%. Die Konfidenzintervalle sind mit einer Größe von unter 2% relativ zum Durchschnitt klein, sodass das Diagramm statistisch auswertbar ist.

Das Diagramm 7.6 zeigt im Kreuzungsszenario, dass die Früherkennung von 4,47s, bei einem *Mindestabstand* von 1s, um 0,49s ansteigt, wenn der *Mindestabstand* auf 2s verdoppelt wird, und entspricht 11%. Für das Auffahrszenario stellt sich eine Verbesserung von 4,31s auf 4,93s ein und entspricht zusätzlichen 0,62s bis zu dem eintreten einer Kollision. Der Anstieg entspricht einem Leistungszuwachs von 14% bei einer Verdoppelung des Abstands. Der Abstand zum vorausfahrenden Automobil wirkt sich auf die Kollisionswahrscheinlichkeit aus, weil die Verringerung der Geschwindigkeit eine exponentielle Auswirkung auf die zurückgelegte Strecke $s(t) = v_0 * t - \frac{1}{2} * a * t^2$ hat, die von t abhängig ist und linear durch den Fahrer beeinflusst wird, indem der Abstand vergrößert wird. Daraus folgt, dass ein *Mindestabstand* von über 2 Sekunden eine relativ größere positive Auswirkung auf die Unfallwahrscheinlichkeit hat, als die Vergrößerung von einer auf zwei Sekunden und vice versa. Diagramm 7.6 zeigt, dass durch einen größeren *Mindestabstand* Kollisionen von vorausfahrenden Automobilen früher erkannt

und länger auf diese reagiert wird. Das wird durch die Gleichung 4.1 im Abschnitt 4.6.4 bewiesen. Durch die Gleichung 4.1 ergibt sich eine Früherkennung einer Bremsung von 3,63, 4,03 und 3,5 und 4,01 vorausfahrender Verkehrsteilnehmer für das Kreuzungs- bzw. Auffahrszenario, wodurch die Anforderung 16 eingehalten wird. Weiterhin wird gezeigt, dass die Implementation des Beschleunigungsverhalten im Abschnitt 5.3.5 die Anforderung 14 im Abschnitt 4.4 erfolgreich einhält.

Iterationsvariable *Positionsungenauigkeit*

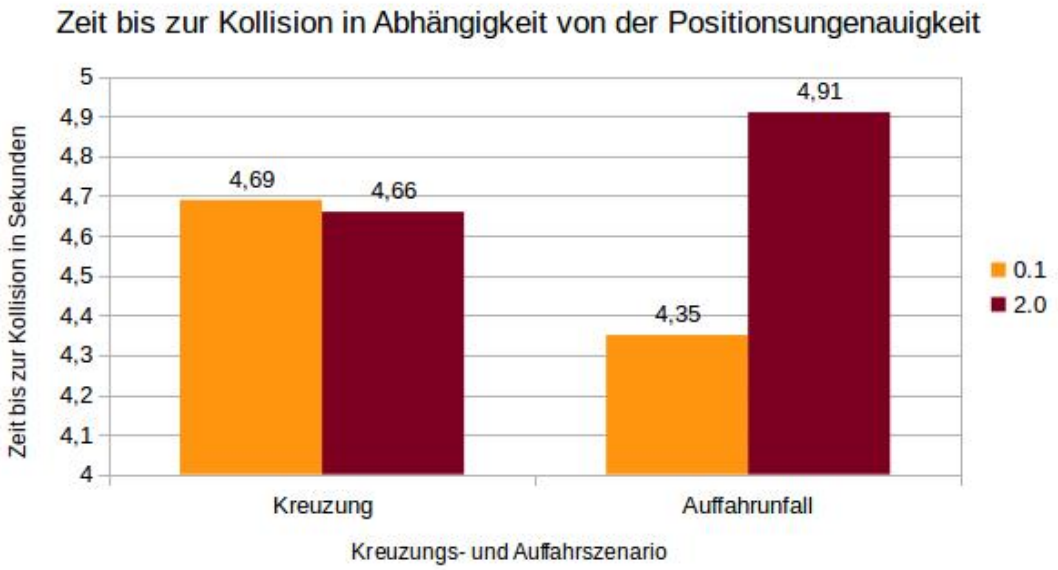


Abbildung 7.7: Das Diagramm zeigt die Zeit bis zur Kollision in Abhängigkeit von der Positionsungenauigkeit.

Das Diagramm 7.7 zeigt die durchschnittliche Zeit pro Verkehrsteilnehmer bis zum Eintreten der Kollision in Abhängigkeit von der *Positionsungenauigkeit*. Wie in den vorherigen Diagrammen wird in Kreuzungs- und Auffahrszenario unterschieden. In der Tabelle 7.8 ist die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistiken im Diagramm 7.7 einzusehen.

Die Tabelle 7.8 zeigt für die Statistiken aus dem Diagramm 7.7 die Stichprobengröße, welche alle in der Größenordnung von 30.000 liegen und den Durchschnitt, der zwischen vier und fünf Sekunden liegt. Für das Auffahrszenario mit der Ungenauigkeit von 2m ist

Name	Stichprobengröße	Durchschnitt	Std. Abweichung	Konfidenzintervall
Kreuzung 0,1m	34068	4,69	2,07	0,02
Kreuzung 2,0m	35006	4,66	2,18	0,02
Heck 0,1	31434	4,35	1,85	0,02
Heck 2,0m	31332	4,91	6,19	0,07

Tabelle 7.8: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik T2CP in Abhängigkeit von der Positionsungenauigkeit in dem Diagramm 7.7.

erkennbar, dass die Verteilung nach links verschoben ist, sodass die Stichproben um den Erwartungswert dicht liegen und nach rechts auslaufen.

Das Diagramm 7.7 zeigt für das Auffahrszenario, dass die Leistungsfähigkeit eine Steigerung um 0,56s von 4,35s auf 4,91s erfährt, an der zuerkennen ist, dass die Anforderung 13 im Abschnitt 4.4 erfüllt ist. Im Kreuzungsszenario stellt sich mit der Verzwanzigfachung der Positionsungenauigkeit eine Leistungsminderung von 0,03 Sekunden ein. Durch die Gleichung 4.1 im Abschnitt 4.6.4 wird gezeigt, dass die Bremsung von Automobilen erkannt wird, die durchschnittlich 3,81, 3,79, und 3,54 und 3,99 Fahrzeuge vorausfahren im Kreuzungs- bzw. Auffahrszenario, wodurch zu erkennen ist, dass die *Positionsungenauigkeit* die Aussage durch das Diagramm 7.6 weiterhin bestätigt, da die *Positionsungenauigkeit* bis 2m in einen größeren Abstand zum direkt vorausfahrenden Automobil resultiert. Auf Basis dieser Argumentation ist die Anforderung 16 im Abschnitt 4.4 eingehalten.

7.4.2 Statistik *Kollisionen*

In diesem Abschnitt sind die Simulationsergebnisse für die Statistik *Kollisionen* dargestellt in Abhängigkeit von den Iterationsvariablen *Mindestabstand*, *Positionsungenauigkeit*, *Reaktionszeit*, *Ausnahmefahrer* und *Sendeintervall*. Die Statistik *Kollisionen* beschreibt die durchschnittliche Kollisionswahrscheinlichkeit für jeden Verkehrsteilnehmer.

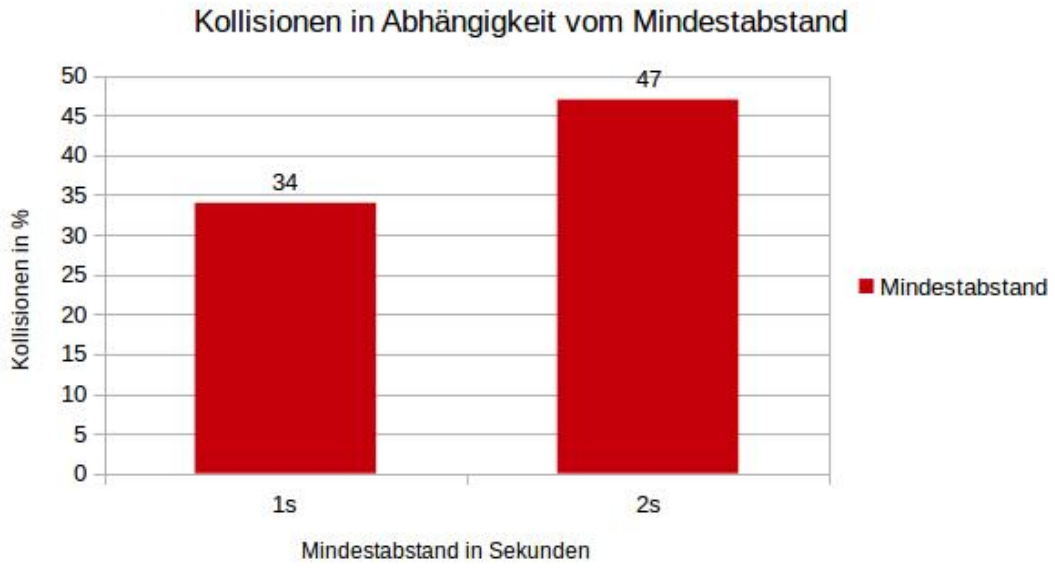


Abbildung 7.8: Das Diagramm zeigt die Entwicklung der Statistik Kollisionen in Abhängigkeit von dem Mindestabstand.

Iterationsvariablen *Mindestabstand* und *Positionsungenauigkeit*

Das Diagramm 7.8 zeigt die durchschnittliche Unfallwahrscheinlichkeit pro Verkehrsteilnehmer in Abhängigkeit vom *Mindestabstand*. Die Tabelle 7.9 zeigt die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die dargestellte Statistik im Diagramm 7.8.

Name	**Stichprobengröße**	**Durchschnitt**	**Std. Abweichung**	**Konfidenzintervall**
1s	54922	34	57,97	0,48
2s	55260	47	65,88	0,55

Tabelle 7.9: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik Kollisionen in Abhängigkeit von dem Mindestabstand in dem Diagramm 7.8.

Die Tabelle 7.9 zeigt für die Statistik im Diagramm 7.8 die Stichprobengröße, welche in der Größenordnung von 50.000 liegt und eine Standardabweichung, welche für beide Statistiken größer als der Erwartungswert ist, sodass die Stichproben, in der nach links

verschobenen Verteilung, weiter auseinander liegen. Das Konfidenzintervall ist für eine und zwei Sekunden mit 1,4% bzw. 1,2% klein, sodass das Diagramm eine statistische Aussagekraft hat.

Das Diagramm 7.8 zeigt die Wahrscheinlichkeit pro Verkehrsteilnehmer an einer Kollision beteiligt zu sein. Dabei ist der *Mindestabstand* von einer und zwei Sekunden im Diagramm aufgetragen. Basierend auf den Simulationsergebnissen in 7.6, bei denen mit steigendem Abstand eine Kollision in beiden Szenarien früher erkannt wird, zeigt die Kollisionsentwicklung keinen zu erwartenden Verlauf im Diagramm 7.8. Bei einer Sekunde ergibt sich eine Unfallwahrscheinlichkeit von 34% und mit der Verdoppelung des Abstandes auf zwei Sekunden eine Unfallwahrscheinlichkeit von 47%, also einer Steigerung von 13%, obwohl ein Rückgang zu erwarten ist.

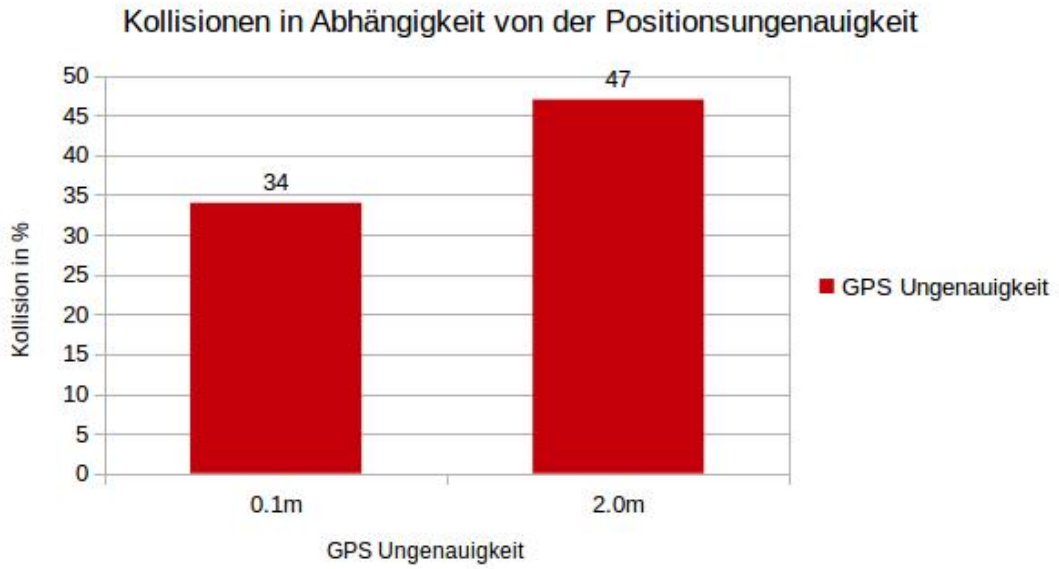


Abbildung 7.9: Das Diagramm zeigt die Entwicklung der Statistik Kollisionen in Abhängigkeit von der Positionsungenauigkeit.

Das Diagramm 7.9 zeigt die durchschnittliche Wahrscheinlichkeit pro Verkehrsteilnehmer an einer Kollision beteiligt zu sein in Abhängigkeit von der *Positionsungenauigkeit*. Die Tabelle 7.10 zeigt die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall der Statistik im Diagramm 7.9.

Name	Stichprobengröße	Durchschnitt	Std. Abweichung	Konfidenzintervall
0,1m	54198	34	56,63	0,48
2,0m	55984	47	66,91	0,55

Tabelle 7.10: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik Kollisionen in Abhängigkeit von der Positionsungenauigkeit in dem Diagramm 7.9.

Die Tabelle 7.10 zeigt für die Statistik im Diagramm 7.9 eine Stichprobengröße mit der Größenordnung von fünfzigtausend und eine Standardabweichung, die größer ist als der Erwartungswert, sodass die Verteilung für beide Statistiken nach links verschoben ist und die Stichproben nach rechts auslaufen. Das Konfidenzintervall ist mit 1,4% bzw. 1,2% vom Erwartungswert klein, sodass das Diagramm eine statistische Aussagekraft hat.

Das Diagramm 7.9 zeigt die Entwicklung der Statistik *Kollisionen* in Abhängigkeit von den *Positionsungenauigkeiten* 0,1m und 2,0m. Das Diagramm 7.6 zeigt, dass die Früherkennung im Auffahrszenario um 0,56s zunimmt, wenn die *Positionsungenauigkeit* auf 2m angehoben wird, wodurch die Anforderung 13 im Abschnitt 4.4 erfüllt ist. Basierend auf dieser Darstellung zeigt die Kollisionsentwicklung im Diagramm 7.9 keinen zu erwartenden Verlauf. Bei einer *Ungenauigkeit der Position* von 0,1m ist die Unfallwahrscheinlichkeit bei 34% und steigt um 13% auf 47%, wenn die *Ungenauigkeit der Position* auf 2m steigt, obwohl eine relative Leistungssteigerung wie im Diagramm 7.7 zu erwarten ist.

Die Diagramme 7.8 und 7.8 zeigt zusammen mit den Diagrammen 7.6 und 7.7, dass das Einfügen der Automobile in die Simulation fehlgeschlagen ist, denn durch einhalten des *Mindestabstands* und den zusätzlichen Abstand, der sich durch die *Positionsungenauigkeit* ergibt, wird in der Anfangsphase vermehrt gebremst, sodass mehr Kollisionen entstanden sind. Der weitere Verlauf weist eine positive Auswirkung auf die Leistungsfähigkeit auf, wie an der Früherkennung von Kollisionen in den Diagrammen 7.6 und 7.7 einzusehen ist. Die Diagramme 7.6 und 7.7 zeigen, dass mit der Vergrößerung des *Mindestabstands* und *Positionsungenauigkeit*[3] Kollisionen früher zu erkennen sind, sodass insgesamt eine positive Entwicklung zu gegeben ist. Aus der Analyse des Diagramms 7.8 und 7.9 kann nicht geschlossen werden, dass der *Mindestabstand* und die *Positionsungenauigkeit* einen

[3]In der Größenordnung, die für diese Ausarbeitung im Kapitel 4.6.1 festgelegt ist

positiven Einfluss auf die Kollisionsentwicklung haben, dennoch ist die Hypothese 2 und 3 im Abschnitt 4.7 durch die Diagramme 7.6 und 7.7 indirekt belegt.

Für die Diagramme 7.9 und 7.8 wird festgestellt, dass das Ansteigen der Kollisionen auf das Einfügen der Automobile in die Simulation zurückzuführen ist, denn durch die *Positionsungenauigkeit* und den größeren *Mindestabstand* wird vermehrt durch die Verkehrsteilnehmer in der Anfangsphase gebremst. Dies wird weiterhin belegt durch die Verkürzung der durchschnittlich gefahrenen Strecke um 49,19m (vgl. Abschnitt 7.2). Tritt ein Auffahrunfall ein wird das auffahrende Automobil an einen freien und voraus liegenden Straßenabschnitt wieder eingefügt (vgl. Kapitel 4.3.2). Weiterhin wird durch die Gleichung 4.1 im Abschnitt 4.6.4 gezeigt, dass die Automobile mit einem Abstand von $1,23\frac{Sekunden}{Fahrzeug}$ in die Simulation eingefügt sind, wodurch weiterhin belegt ist, dass das Einfügen der Automobile in die Simulation fehlgeschlagen ist. Basierend auf dieser Argumentation und die im Diagrammen 7.6 und 7.7 positiv dargestellte Entwicklung der Früherkennung, wenn der *Mindestabstand* größer wird und die *Positionsungenauigkeit* auf 2m ansteigt, wird im folgenden für die Statistik *Kollisionen*, die um 26% verringerte Kollisionswahrscheinlichkeit, in Klammern hinter den Ergebnissen der Diagramme angegeben, weil der *Mindestabstand* und die *Positionsungenauigkeit* die Unfallwahrscheinlichkeit durchschnittlich je um 13% erhöht.

Iterationsvariable *Reaktionszeit*

Das Diagramm 7.10 zeigt den Einfluss der *Reaktionszeit* auf die durchschnittliche Kollisionswahrscheinlichkeit pro Verkehrsteilnehmer. Für das Diagramm 7.10 ist in Tabelle 7.11 die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall einzusehen.

Name	**Stichprobengröße**	**Durchschnitt**	**Std. Abweichung**	**Konfidenzintervall**
0,6s	55381	39	61,4	0,511
1,0s	54801	42	63,4	0,531

Tabelle 7.11: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik Kollisionen in Abhängigkeit von der Reaktionszeit des Fahrers in dem Diagramm 7.10.

Die Tabelle 7.11 zeigt, dass die Statistiken im Diagramm 7.10 eine Stichprobengröße von über 50.000 aufweisen mit einem Konfidenzintervall von je 1,3%, sodass die Stichprobengröße als ausreichend groß bestätigt ist. Weiterhin ist zu erkennen, dass die

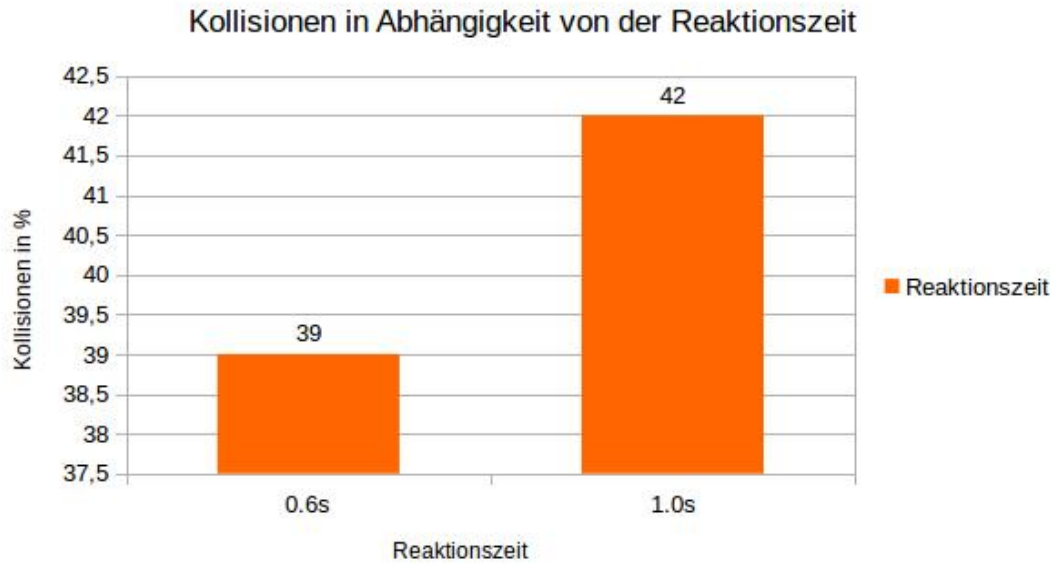


Abbildung 7.10: Das Diagramm zeigt die Statistik Kollisionen in Abhängigkeit von der Reaktionszeit des Fahrers.

Standardabweichung größer ist als der Erwartungswert, sodass die Stichproben die kleiner als der Durchschnitt sind, in der Verteilung konzentrierter liegen als die Stichproben größer als der Durchschnitt.

Das Diagramm 7.10 zeigt, dass die Reaktionssteigerung von 0,4 Sekunden die Kollisionswahrscheinlichkeit um 3% von 42(16)% auf 39(13)% beeinflusst. Die *Reaktionszeit* wirkt sich auf die Kollisionswahrscheinlichkeit aus, weil die Verringerung der Geschwindigkeit eine exponentielle Auswirkung auf die zurückgelegte Strecke $s(t) = v_0 * t - \frac{1}{2} * a * t^2$ hat, die von t abhängig ist und linear durch eine frühere Bremsung beeinflusst wird, woraus folgt, dass mit der *Reaktionszeit* unterhalb von 0,6 Sekunden die Kollisionswahrscheinlichkeit mehr abnimmt, als zwischen 1,0 und 0,6 Sekunden und vice versa. Basierend auf den Simulationsergebnissen die im Diagramm 7.10 graphisch dargestellt sind und der durchgeführten Analyse ist die Hypothese 4 im Abschnitt 4.7 belegt.

Iterationsvariable *Ausnahmefahrer*

Das Diagramm 7.11 zeigt den Verlauf der Statistik *Kollisionen* in Abhängigkeit von dem prozentualen Anteil der *Ausnahmefahrern* im Szenario. Die Tabelle 7.12 zeigt

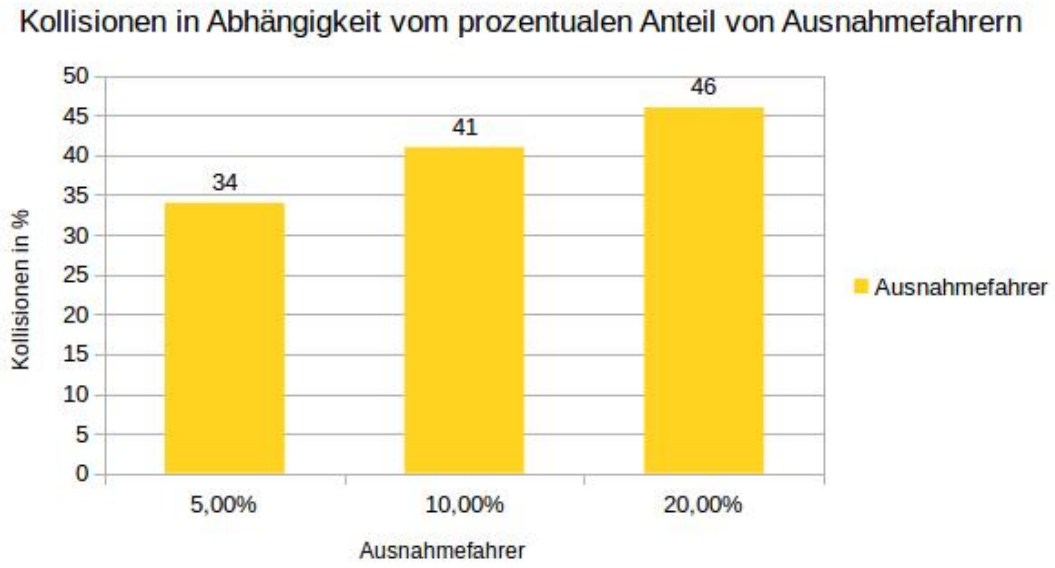


Abbildung 7.11: Das Diagramm zeigt den Verlauf der Kollisionen in Abhängigkeit von der Anzahl der Ausnahmefahrer.

für die Statistik Kollisionen im Diagramm 7.11 die Stichprobengröße, Durchschnitt, Standardabweichung und das Konfidenzintervall.

Name	**Stichprobengröße**	**Durchschnitt**	**Std. Abweichung**	**Konfidenzintervall**
5%	36584	34	58,7	0,60
10%	36745	41	62,1	0,63
20%	36853	46	65,5	0,67

Tabelle 7.12: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik Kollisionen in Abhängigkeit von der Anzahl der Ausnahmefahrer in dem Diagramm 7.11.

Die Tabelle 7.12 zeigt, dass die Statistik aus dem Diagramm 7.11 eine Stichprobengröße zwischen 30.000 und 40.000 aufweist, mit einem Konfidenzintervall zwischen 1% und 1,7%. Die Standardabweichung der drei Statistiken zeigen zusammen mit dem Durchschnitt, dass die Verteilung nach links verschoben ist. Die Stichproben größer als der Durchschnitt verteilen sich stärker rechts vom Erwartungswert.

Das Diagramm 7.11 zeigt für einen Anteil von 20% *Ausnahmefahrern* eine Kollisionswahrscheinlichkeit von 46(20)% und für eine Halbierung der *Ausnahmefahrer* auf 10% eine Verringerung von 5% auf 41(15)%, welches einem Leistungszuwachs von 11% entspricht. Für eine weitere Halbierung der *Ausnahmefahrer* auf 5% nimmt die Kollisionswahrscheinlichkeit um weitere 7% auf 34(8)% ab und entspricht einem Leistungszuwachs von 17,1%. Der Leistungszuwachs von 10% auf 5% *Ausnahmefahrer* ist um 155,4% größer als der von 20% auf 10% *Ausnahmefahrer*, sodass eine exponentielle Leistungssteigerung des Assistenzsystems zu erkennen ist, mit einer linear zunehmenden Anzahl von Verkehrsteilnehmer mit aktivem Assistenzsystem. Basierend auf den Simulationsergebnissen, die im Diagramm 7.11 graphisch dargestellt sind und der durchgeführten Analyse ist die Hypothese 5 aus dem Abschnitt 4.7 belegt.

Iterationsvariable *Sendeintervall*

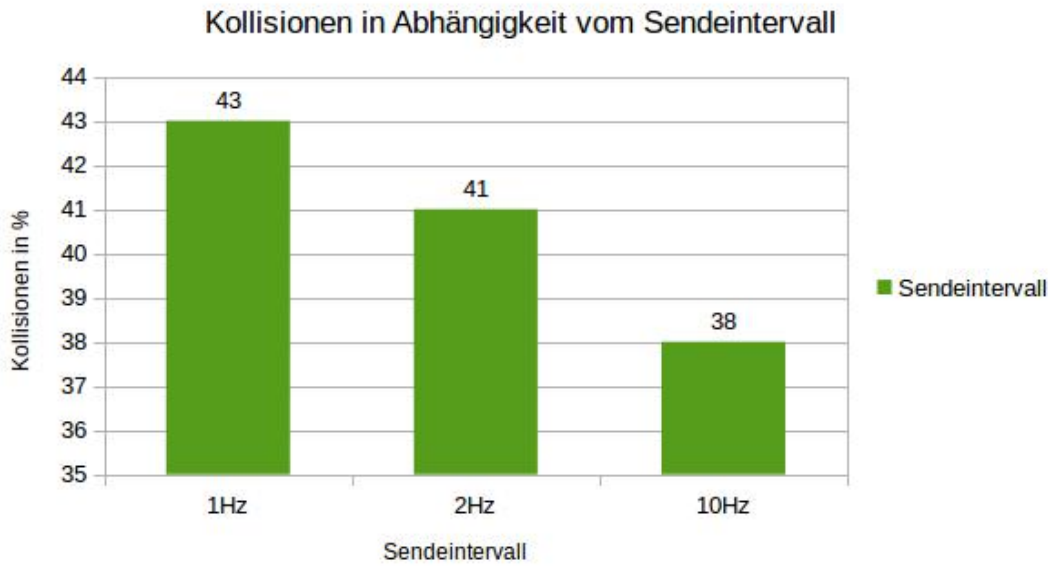


Abbildung 7.12: Das Diagramm zeigt die Entwicklung der Kollisionen in Abhängigkeit von dem Sendeintervall.

Das Diagramm 7.12 zeigt den Einfluss des *Sendeintervalls* auf die durchschnittliche Kollisionswahrscheinlichkeit pro Verkehrsteilnehmer. Die Tabelle 7.13 zeigt für die Statistik

Kollisionen im Diagramm 7.12 die Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall.

Name	Stichprobengröße	Durchschnitt	Std. Abweichung	Konfidenzintervall
1Hz	37058	43	61,3	0,62
2Hz	36852	41	63,5	0,65
10Hz	36272	38	62,3	0,64

Tabelle 7.13: Stichprobengröße, Durchschnitt, Standardabweichung und Konfidenzintervall für die Statistik Kollisionen in Abhängigkeit von dem Senderintervall in dem Diagramm 7.12.

Die Tabelle 7.13 zeigt eine Stichprobengröße von über 36000 und ein Konfidenzintervall zwischen 1,4% und 1,6% relativ zum Durchschnitt. Die Standardabweichung zeigt die Streuung der Stichproben in der Verteilung. Die Standardabweichung in Tabelle 7.13 zeigt zusammen mit dem Durchschnitt, dass die Verteilung nach links verschoben ist und die Stichproben kleiner als der Erwartungswert eng beieinander liegen und die Stichproben größer als der Erwartungswert verstreut sind.

Das Diagramm 7.12 zeigt für ein *Sendeintervall* von 1Hz eine Kollisionswahrscheinlichkeit von 43(17)% und mit der Verdoppelung auf 2Hz eine Abnahme um 2% auf 41(15)% und entspricht einer Leistungssteigerung von 4,7%. Mit der Verfünffachung des *Sendeintervalls* von 2Hz auf 10Hz verbessert sich die Kollisionswahrscheinlichkeit um 3% auf 38(12)% und entspricht 7,3%. Es ist zu erkennen, dass der relative Leistungszuwachs von 2Hz auf 10Hz nicht in der Größenordnung von dem Zuwachs von 1Hz auf 2Hz ist. In dem Diagramm 7.1 ist ausgearbeitet, dass die fehlerhaften Broadcast für ein 10Hz Intervall zu nehmen im Vergleich zu 1Hz und 2Hz, außerdem steigt das Verhältnis von $\frac{WS}{TS}$ mit steigendem *Sendeintervall* an, wie für die Diagramme 7.2, 7.3 und 7.3 ausgearbeitet, sodass statistisch nachgewiesen ist, dass die Leistungsfähigkeit des Assistenzsystems im 10Hz Intervall durch die ineffiziente Nutzung des Übertragungsmedium limitiert ist. In dem Diagramm 7.1 ist zu erkennen, dass für ein größeres *Sendeintervall* die absolute Anzahl an fehlerfrei empfangenen Broadcast-Nachrichten steigt und im Diagramm 7.5 ist zu sehen, dass die Früherkennung von Kollisionen mit dem erhöhen des *Sendeintervalls* zunimmt. Zusammen mit dem zuvor ausgearbeiteten Verhalten, dass die Kollisionen mit steigendem *Sendeintervall* abnimmt, welches das Diagramm 7.12 beschreibt, ist die Hypothese 1 aus dem Abschnitt 4.7 belegt. Mit einem MAC Algorithmus, der den Channel-Zugriff für ein 10Hz *Sendeintervall* identisch effizient bereitstellt, wie für das

1Hz und 2Hz *Sendeintervall*, wird durch die Hypothese 1 die Kollisionswahrscheinlichkeit von 43(17)%, 41(15)% und 38(12)% auf 43(17)%, 41(15)% und 28(2)% verbessert.

7.4.3 Bewertung

Dieser Abschnitt hat gezeigt, dass ein Bremsvorgang von Verkehrsteilnehmern erkannt wird die zwischen 3,39 und 4,03 Automobile vorausfahren, sodass der Fahrzeugführer frühzeitig auf ein Abbremsen reagiert und sich ein durchschnittlicher Abstand von 4,17s bis 4,96s zu dem Verkehrsteilnehmer ergibt, für den die Geschwindigkeit angepasst wird. Daraus ergibt sich eine Kollisionswahrscheinlichkeit von 34(8)% bis 46(20)%. In den Diagrammen 7.12 und 7.5 ist zu erkennen, dass die Leistungsfähigkeit des Assistenzsystems durch den MAC-Algorithmus stark limitiert ist. Der MAC-Algorithmus STDMA (vgl. Abschnitt 2.2.2) wird den Zugriff auf das Übertragungsmedium effizienter bereitstellen, da STDMA alle Anforderungen an den MAC-Algorithmus erfüllt, wie im Konzept 5.2.2 ausgearbeitet, sodass eine Kollisionswahrscheinlichkeit von 43(17)%, 41(15)% und 28(2)% für 1Hz, 2Hz bzw. 10Hz möglich ist (vgl. 7.12).

Das Diagramm 7.7 zeigt, dass das Assistenzsystem die *Positionsungenauigkeit* kompensiert und für das Auffahrszenario eine Kollision durchschnittlich um 0,56 Sekunden früher erkennt, während das Verhalten an der Kreuzung auf dem selben Level bleibt. Es hat sich gezeigt, dass großes Potenzial in einer genauen Analyse der empfangenen Daten besteht, sodass komplexe Situationen wie auf einer Kreuzung effizienter aufgelöst werden, wobei mit steigender *Positionsungenauigkeit* der Anspruch steigt.

Dieser Abschnitt hat die Leistungsfähigkeit des Assistenzsystems ausgearbeitet und welchen Einfluss externe Parameter auf die Kollisionswahrscheinlichkeit haben. Mit Kenntnis der Parameter wie die persönliche *Reaktionszeit*, dem *Sendeintervall* der verwendeten Hardware, den Verkehrsteilnehmern ohne Assistenzsystem und *Positionsungenauigkeit* oder einer Teilmenge davon, kann der *Mindestabstand* dem Sicherheitsverlangen des Fahrers angepasst werden, sodass eine Strecke mit höchster Sicherheit langsamer zurück gelegt wird oder schneller ohne dass die Sicherheit abnimmt. Ältere Menschen mit einer verminderten Reaktionsfähigkeit können durch Vergrößern des *Mindestabstands* das Defizit ausgleichen und ihrem Sicherheitsverlangen entsprechend das Automobil fahren, wohingegen jüngere Verkehrsteilnehmer mit überdurchschnittlicher Reaktionszeit und kleinerem Mindestabstand bei unveränderter Sicherheit früher ihr Ziel erreichen. Außerdem wird die Leistung gesteigert, wenn der Fahrzeugführer aktuelle Technologie verwendet, die schneller und genauer die Position aktualisiert.

Dieses Kapitel hat die Hypothesen 1,4, und 5 im Kapitel 4.7 belegt. Die Statistik *Kollisionen* weist die Leistungsfähigkeit des Assistenzsystems nach, welche durch die Früherkennung von Kollisionen definiert ist. Das heißt, dass das Sinken der Kollisionen eine hinreichende Bedingung für die Früherkennung der Kollisionen ist, sodass die Hypothesen 2 und 3 nicht belegt sind. Weiterhin sind in diesem Kapitel die Anforderungen 13, 14, 15 und 16 zusätzlich erfolgreich bestätigt.

8 Zusammenfassung, Fazit und Ausblick

Das Ende dieser Arbeit teilt sich in die drei Abschnitte Zusammenfassung, Fazit und Ausblick auf. In dem Abschnitt Zusammenfassung wird diese Ausarbeitung und die Ergebnisse zusammengefasst dargestellt, sodass im Abschnitt Fazit das entwickelte Assistenzsystem auf Basis der Forschungsergebnisse, unter den Gesichtspunkten der Leistungsfähigkeit und Anwendbarkeit, eingeordnet wird. Abschließend wird im Abschnitt Ausblick dargestellt welche zukünftigen Forschungsmöglichkeiten durch diese Ausarbeitung gegeben sind, sodass eine Einschätzung über den zukünftigen Stellenwert getroffen wird.

8.1 Zusammenfassung

Für diese Ausarbeitung wurde ein Assistenzsystem entwickelt und die Leistungsfähigkeit simulationsbasiert erforscht. Dafür sind im Kapitel 4 16 Anforderungen definiert, die als Qualitätsstandard für die Ausarbeitung gelten. Die Anforderungen unterteilen sich auf die Teilbereiche Simulation, Lokalisierung, Protokoll Stack und Assistenzsystem. Im Abschnitt 4.5 sind Arbeitspakete definiert, die für diese Ausarbeitung zu bearbeiten sind und im Kapitel 6 umgesetzt wurden. Weiterhin sind im Kapitel 4.7 Hypothesen definiert, die durch diese Ausarbeitung zu belegen sind. Die Hypothesen machen Annahmen über das Verhalten der entstandenen Kollisionen in Abhängigkeit von den Einflüssen *Sendeintervall*, *Reaktionszeit*, *Ausnahmefahrer*, *Mindestabstand* und *Positionsungenauigkeit*.

Es hat sich gezeigt, dass die Anforderungen 1-5 an die Simulation eingehalten werden. Im Kapitel 5 wird das Konzept für die Lokalisierung, Protokoll Stack, Assistenzsystem und Statistik ausgearbeitet, mit der die Implementation im Kapitel 6 stattfindet. Es hat sich gezeigt, dass die ausgearbeitete Lokalisierung die Anforderungen 6-8 einhält. Die Anforderungen 9-12 beschreiben die Qualitätsstandards an den Protokoll Stack. Es wurde ausgearbeitet, dass STDMA alle Anforderungen an den MAC-Algorithmus erfüllt und CSMA/CA überlegen ist. Nichtsdestotrotz stellt das INET Framework das STDMA Protokoll nicht zur Verfügung, sodass der durch Veins bereitgestellte 802.11p Protokoll Stack verwendet wurde, der die Anforderungen 10 und 11 nicht einhält.

Im Kapitel 6 wird gezeigt, dass das Assistenzsystem die an ihn gestellten Anforderungen 13-15 implementiert. Durch die im Kapitel 7 dargestellten Simulationsergebnisse wird gezeigt, dass das Assistenzsystem alle Anforderungen erfüllt, sodass belegt ist, dass das Konzept für das Assistenzsystem im Kapitel 5 und die Implementierung im Abschnitt 6 erfolgreich waren.

Durch die Diagramme 7.11, 7.12 und 7.10 worden die Hypothesen 1,4 und 5 bewiesen. Weiterhin hat sich gezeigt, dass aufgrund des *Mindestabstands* und *Positionsungenauigkeit* das Einfügen der Automobile in die Simulation fehlerbehaftet ist, sodass durch verstärktes bremsen die Statistiken 7.8 und 7.9 durch künstlich entstandene Kollisionen verfälscht sind. Die Diagramme 7.6 und 7.7 haben gezeigt, dass die Variablen *Positionsungenauigkeit* und *Mindestabstand* eine positive Auswirkung auf die Kollisionsfrüherkennung haben, sodass die Hypothesen 2 und 3 nur indirekt bewiesen sind.

Die Simulationsergebnisse haben gezeigt, dass 68,2% der Verkehrsteilnehmer durchschnittlich mit einer Geschwindigkeit zwischen $\frac{10{,}93m}{s}$ und $\frac{8{,}06m}{s}$ durch das Szenario fahren und realistisch für ein Stadtszenario ist.

Die Analyse des Protokoll Stacks bestätigt die Defizite des CSMA/CA MAC Algorithmus, die im Konzept 5 ausgearbeitet sind. Die Simulation hat gezeigt, dass im Backoff Algorithmus 21,22%-48% der Broadcasts verworfen werden, jede Nachricht vor dem Senden durchschnittlich 3,79-4,06 Slots warten muss und von den gesendeten Nachrichten 8,5%-20% mit Bit-Error empfangen werden.

Für das Assistenzsystem wurde gezeigt, dass auf das Verhalten von Verkehrsteilnehmer reagiert werden kann, die durchschnittlich 4,03 Automobile vorausfahren, sodass eine Kollisionswahrscheinlichkeit von 8%-20% möglich ist, die durch das STDMA Protokoll auf 2%-15% verbessert werden könnte.

8.2 Fazit

Diese Ausarbeitung hat gezeigt, dass das entwickelte Assistenzsystem Potenzial hat im Straßenverkehr Kollisionen zu vermeiden. Durch die Früherkennung von Kollisionen wird die Schrecksekunde kompensiert, weil der Fahrzeugführer durch das Assistenzsystem frühzeitig auf ein bevorstehendes Ereignis hingewiesen wird. Außerdem steigt die Leistungsfähigkeit des Assistenzsystem exponentiell mit der Anzahl der Nutzer. Als Nachteil ist zu erkennen, dass die Funktionalität ausschließlich von der Anzahl der Nutzer abhängig ist, sodass eine Markteinführung als „Stand-Alone" unwahrscheinlich ist.

Diese Arbeit eröffnet weitere Forschungsmöglichkeiten in der Analyse der zur Verfügung stehenden Daten und dem optimierten senden der Broadcasts in Abhängigkeit von der Gefährdung des Automobils, sodass die Ausnutzung des Übertragungsmediums optimiert wird.

8.3 Ausblick

Es ist zu erkennen, dass trotz nachgewiesener Funktionalität die Anwendbarkeit als „Stand-Alone" unzureichend ist, sodass ein Signal-Combining die Anwendbarkeit und Verlässlichkeit erhöht. Wenn dem Assistenzsystem Messwerte aus Laserscannern, Sensoren und Kameras zur Verfügung stehen, wird auf Basis unterschiedlicher Datenquellen die Verlässlichkeit des Assistenzsystems erhöht.

Großes Potenzial ist in der Analyse der zur Verfügung stehenden Daten zu erkennen. Durch Maschinen Lernen kann das Assistenzsystem aus den zur Verfügung stehenden Daten und den daraus resultierenden Ereignissen die Funktionalität stark verbessern. Dadurch ist es möglich, dass das Assistenzsystem eine Kollision mit einer Wahrscheinlichkeit vorhersagt, bevor diese eingetreten ist. Weiterhin können komplexe Kreuzungssituationen effizienter und kollisionsfreier aufgelöst werden, da das Assistenzsystem das Verhalten des Systems präziser voraussagt.

Diese Ausarbeitung stellt einen kleinen Schritt in Richtung autonom fahrender Automobile da, dennoch ist zu erkennen, dass ein großes Potenzial in diesem Bereich ist. Die Mobilität der Gesellschaft wird in naher Zukunft durch autonom fahrende Automobile revolutioniert. LKW Transporte und Öffentliche Verkehrsmittel können durch fahrerlose und autonom fahrende Automobile wesentlich effizienter durchgeführt werden. Durch Car-Sharing können autonom fahrende „Taxis" bestellt werden und eine Gruppe von Personen befördern. Geschäftsmänner können die Fahrzeit im Automobil als Büro effizient ausnutzen. Außerdem wird die Jugend im autonomen Fahrzeug am Straßenverkehr teilnehmen.

Literaturverzeichnis

[1] Statistisches Bundesamt, Zahl der Verkehrstoten 2015 um 2,9 % gestiegen, URL https://www.destatis.de/DE/PresseService/Presse/%Pressemitteilungen/2016/02/PD16_060_46241.html, Abruf 10.05.2016

[2] Bundesanstalt für Straßenwesen, Volkswirtschaftliche Unfallkostenrechnung, URL http://www.bast.de/DE/Verkehrssicherheit/Fachthemen/u1-unfallkostenrechnung/u1-unfallkostenrechnung.htm, Abruf 10.05.2016

[3] Veins Handbuch, URL http://veins.car2x.org/documentation/, Abruf 22.07.2016

[4] Veins Module, URL http://veins.car2x.org/documentation/modules/, Abruf 22.07.2016

[5] Christoph Sommer, David Eckhoff, Reinhard German, A computationally inexpensive empirical model of IEEE 802.11p radio shadowing in urban environments (2011), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=5720204, Abruf 22.07.2016

[6] Christoph Sommer, David Eckhoff, Reinhard German, IVC in Cities: Signal Attenuation by Buildings and How Parked Cars Can Improve the Situation (2014), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6544519, Abruf 22.07.2016

[7] SUMO Einführung, URL http://sumo.dlr.de/wiki/Sumo_at_a_Glance, Abruf 23.07.2016

[8] SUMO Zufallswerte, URL http://sumo.dlr.de/wiki/Simulation/Randomness, Abruf 23.07.2016

[9] SUMO Car Following Models, URL http://sumo.dlr.de/wiki/Definition_of_Vehicles,_Vehicle_Types,_and_Routes#Car-Following_Models, Abruf 23.07.2016

[10] SUMO Lane Changing Models, URL http://sumo.dlr.de/wiki/Definition_of_Vehicles,_Vehicle_Types,_and_Routes#Lane-Changing_Models, Abruf 23.07.2016

[11] Mozilla Location Service, https://location.services.mozilla.com/, Abruf 11.05.16

[12] Martin Harbich, GPS wie funktioniert's Ausarbeitung (2010), URL http://www.cs.hs-rm.de/~linn/fachsem0910/harbich/GPS_Ausarbeitung.pdf, Abruf 11.05.2016

[13] William J. Hughes, Global Positioning System (GPS) Standard Positioning Service (SPS) Performance Analysis Report (2014), URL http://www.nstb.tc.faa.gov/reports/PAN86_0714.pdf, Abruf 11.05.2016

[14] Felix Liedeker, Relativitätstheorie und GPS (2013), URL http://www.physik.uni-bielefeld.de/~yorks/pro13/liedeker.pdf, Abruf 11.05.16

[15] Katrin Bilstrup, Elisabeth Uhlemann, Erik G. Ström, Evaluation of the IEEE 802.11p MAC Method for Vehicle-to-Vehicle Communication (2008), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=4657278, Abruf 12.05.16

[16] WI-FI Direct - was ist das eigentlich?, URL http://wifi-direct.net/2011/06/23/wi-fi-direct-was-ist-das-eigentlich/, Abruf 15.05.2016

[17] Bluetooth Range, URL http://www.bluair.pl/bluetooth-range, 12.05.2016

[18] Jay A. Farrell, Tony D. Givargis, Matthew J. Barth, Real-Time Differential Carrier Phase GPS-Aided INS (2000), URL http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=852915, Abruf 01.06.2016

[19] C. Mbarushimanaa, A. Shahrabib, T. Buggyb, A cross-layer TCP enhancement in QoS-aware mobile ad hoc networks (2013), URL http://www.sciencedirect.com/science/article/pii/S1389128612003386, Abruf 01.06.2016

[20] T. Gopinath, A.S.Rathan Kumar, Rinki Sharma, Performance Evaluation of TCP and UDP over Wireless Ad-hoc Networks with Varying Traffic Loads (2013), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6524403, Abruf 01.06.2016

[21] Yi Wu, Kenneth W. Shum, Wing Shing Wong, Safety-Message Broadcast in Vehicular Ad Hoc Networks Based on Protocol Sequences (2014), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6587610, Abruf 12.05.16

[22] Open Street Map, URL https://www.openstreetmap.org, Abruf 25.07.2016

[23] SUMO Netconvert, URL http://sumo.dlr.de/wiki/NETCONVERT, Abruf 25.07.2016

[24] Hello SUMO Tutorial, URL http://sumo.dlr.de/wiki/Tutorials/Hello_Sumo, Abruf 25.07.2016

[25] Einführung in TraCI, URL http://sumo.dlr.de/wiki/TraCI, Abruf 25.07.2016

[26] Standard Car Following Modell in SUMO http://sumo.dlr.de/wiki/Definition_of_Vehicles,_Vehicle_Types,_and_Routes#Default_Krauss_Model_Description, Abruf 28.09.2016

[27] Stefan Krauß, Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics (1998), URL http://sumo.dlr.de/pdf/KraussDiss.pdf, Abruf 28.09.2016

[28] Digitale Kommunikation und Internet Dienste, URL http://www.rvs.uni-bielefeld.de/lecture/DKI/W04/ws200405_dki1_termin4.pdf, Abruf 28.09.2016

[29] GPS Gloassary, URL https://learn.sparkfun.com/tutorials/gps-basics/gps-glossary, Abruf 28.09.2016

[30] Technische Spezifikation vom Samsung Galaxy S5, URL http://www.samsung.com/de/consumer/mobile-devices/smartphones/galaxy-s/SM-G900FZKADBT, Abruf 28.09.2016

[31] Salih, Zaini, Zhahir, The Suitability of GPS Receivers Update Rates for Navigation Applications, URL http://waset.org/publications/7783/the-suitability-of-gps-receivers-update-rates-for-navigation-applications, Abruf 28.09.2016

[32] Henrik Zöller, Wolfgang Hugemann, Zur Problematik der Bremsreaktionszeit im Straßenverkehr, URL http://www.unfallrekonstruktion.de/pdf/bdp_1998_german.pdf, Abruf 28.09.2016

[33] Error analysis for the Global Positioning System, URL https://en.wikipedia.org/wiki/Error_analysis_for_the_Global_Positioning_System, Abruf 28.09.2016

[34] Reuters, EU launches free satellite system to fine-tune GPS, URL http://www.reuters.com/article/us-eu-transport-satellite-idUSTRE59023F20091001, Abruf 28.09.2016

[35] Frank E. Ritter, Michael J. Schoelles, Karen S. Quigley, Laura Cousino Klein, Determining the number of simulation runs: Treating simulations as theories by not sampling their behavior (2011), URL http://acs.ist.psu.edu/papers/ritterSQKip.pdf, Abruf 28.09.2016

[36] Mercedes C Klasse und VW Passat im Test, URL http://www.autobild.de/artikel/vw-passat-mercedes-c-klasse-kombi-vergleich-5599595.html, Abruf 28.09.2016

[37] Bremsverzögerung und Hochrechnung, URL ftp://internic.at/Druckluftbremse/8150200573-23.pdf, Abruf 28.09.2016

[38] Verkehrssituation auf Deutschlands Straßen, URL https://www.bundesregierung.de/Content/DE/Magazine/MagazinInfrastrukturNeueLaender/014/s1-verkehrssituation-auf-deutschlands-strassen.html, Abruf 28.09.2016

[39] Manuelle Straßenverkehrszählung 2010, Ergebnisse auf Bundesstraßen, URL http://www.bast.de/DE/Statistik/Verkehrsdaten-Downloads/2010/zaehlung-2010-bundesstrassen.pdf, Abruf 28.09.2016

[40] Global Positioning System (GPS), Standard Positioning Service (SPS), Performance Analysis Report (2014), URL http://www.nstb.tc.faa.gov/reports/PAN86_0714.pdf, Abruf 07.10.2016

[41] Is Military GPS More Accurate Than Civilian GPS?, URL http://www.gps.gov/systems/gps/performance/accuracy/, Abruf 07.10.2016

[42] SUMO Speed Distributions, URL http://sumo.dlr.de/wiki/Definition_of_Vehicles,_Vehicle_Types,_and_Routes#Speed_Distributions, Abruf 07.10.2016

[43] Definition der Simulation, URL http://wirtschaftslexikon.gabler.de/Definition/simulation.html, Abruf 09.10.2016

[44] OMNeT++ Community: OMNeT++ 5.0, URL https://omnetpp.org/, Abruf 10.10.2016

[45] OMNeT++ Simulation Manual, URL https://omnetpp.org/doc/omnetpp/manual/, Abruf 10.10.2016

[46] Simulation of Urban Mobility (SUMO), URL http://www.dlr.de/ts/en/desktopdefault.aspx/tabid-9883/16931_read-41000/, Abruf 11.10.2016

[47] OSI Modell, URL http://www.elektronik-kompendium.de/sites/kom/0301201.html, Abruf 11.10.2016

[48] Sebastian Gräfling, Petri Mähönen, Janne Riihijärvi, Performance evaluation of IEEE 1609 WAVE and IEEE 802.11p for vehicular communications (2010), URL http://ieeexplore.ieee.org/document/5547184/, Abruf 12.10.2016

[49] Marcus Burton, 802.11 Arbitration (2009), URL https://www.cwnp.com/uploads/802-11_arbitration.pdf, Abruf 13.10.2016

[50] Miquel Oliver, Ana Escudero, Mobile Communications Research Group, Study of different CSMA/CA IEEE 802.11-based implementations (1999), URL http://www.eunice-forum.org/eunice99/027.pdf, Abruf 13.10.2016

[51] Andrew S. Tanenbaum, David J. Wetherall, Computer Networks Fifth Edition (2011), ISBN-13: 978-0-13-212695-3, URL https://montcs.bloomu.edu/Readings/Computer%20Networks%20-%20A%20Tanenbaum%20-%205th%20edition.pdf, Abruf 13.10.2016

[52] Katrin Bilstrup, Elisabeth Uhlemann, Erik G. Strom, Evaluation of the IEEE 802.11p MAC Method for Vehicle-to-Vehicle Communication (2008), URL http://ieeexplore.ieee.org/document/4657278/, Abruf 14.10.2016

[53] Muhammed Enes Bayrakdar, Ali Çalhan, Delay characteristics of TDMA medium access control protocol for cognitive radio networks (2016), URL http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7589870, Abruf 14.10.2016

[54] Christoph Sommer, Stefan Joerer and Falko Dressler, On the Applicability of Two-Ray Path Loss Models for Vehicular Network Simulation (2012), URL http://www.ccs-labs.org/bib/sommer2012applicability/sommer2012applicability.pdf, Abruf 15.10.2016

[55] Christoph Sommer, Falko Dressler, Using the Right Two-Ray Model? A Measurement-based Evaluation of PHY Models in VANETs (2011), URL http://www.ccs-labs.org/bib/sommer2011using/sommer2011using.pdf, Abruf 15.10.2016

[56] Anleitung für die Implementation selbstentwickelter Car Following Modell, URL http://sumo.dlr.de/wiki/Developer/How_To/Car-Following_Model, Abruf 17.10.2016

[57] INET Dokumentation, URL https://inet.omnetpp.org/Introduction.html, Abruf 18.10.2016

[58] Veins FAQ, URL http://veins.car2x.org/documentation/faq/, Abruf 15.10.2016

[59] Change Log INET, URL https://github.com/inet-framework/inet/blob/v2.3.0/WHATSNEW, Abruf 18.10.2016

[60] Veins 4.4 komptatibilität zu INET, URL https://github.com/sommer/veins/blob/veins-4.4/configure#L17, Abruf 18.10.2016

[61] Hujun Yin, Siavash Alamouti, OFDMA: A Broadband Wireless Access Technology (2006), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=4534773, Abruf 20.10.2016

[62] IEEE 802.11p Spezifikation, URL https://www.ietf.org/mail-archive/web/its/current/pdfqf992dHy9x.pdf, Abruf 21.10.2016

[63] Quasi-Zenith Satellite System, URL http://qz-vision.jaxa.jp/USE/en/, Abruf 22.10.2016

[64] Chung-Ming Huang, Shih-Yang Lin, An Advanced Vehicle Collision Warning Algorithm over the DSRC Communication Environment: An Advanced Vehicle Collision Warning Algorithm (2013), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=6531822, Abruf 01.06.16

[65] R. Tatchikou, S. Biswas, F. Dion, Cooperative vehicle collision avoidance using inter-vehicle packet forwarding (2005), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=1578262, Abruf 01.06.16

[66] X. Yang, L. Liu, N.H. Vaidya, A Vehicle-to-Vehicle Communication Protocol for Cooperative Collision Warning (2004), URL ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=1331717, Abruf 01.06.16

[67] Takahiro Wada, Shun'ichi Doi, Shoji Hiraoka, A Deceleration Control Method of Automobile for Collision Avoidance based on Driver?s Perceptual Risk (2009), URL http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5354640, Abruf 01.06.16

[68] Tsen-Wei Chang, Jau-Woei Perng, Design and Implementation of the Intelligent Stop and Go System in Smart Car, TAIWAN iTS-1 (2006), URL http://ieeexplore.ieee.org/xpls/icp.jsp?arnumber=4274172, Abruf 01.06.16

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 14. November 2016 Janek Tichy