Masterarbeit

Co-Simulationsbasierter Ansatz zur Entwicklung aktiver Sicherheitssysteme

Markus Schratter

Institut für Elektrische Meßtechnik und Meßsignalverarbeitung
Technische Universität Graz
Vorstand: Univ.-Prof. Dipl.-Ing. Dr. techn. Georg Brasseur

Begutachter: Univ.-Doz. Dipl.-Ing. Dr. techn. Daniel Watzenig

Betreuer: Univ.-Doz. Dipl.-Ing. Dr. techn. Daniel Watzenig

Betreuer: Dr. Daniel Schwarz (BMW Group)

Graz, im August 2013

Kurzfassung

Aufgrund stetiger Weiterentwicklung im Bereich der Sensorik und zuverlässiger Signalverarbeitung kommt es zu wesentlichen Fortschritten in der Integralen Fahrzeugsicherheit. Bei der Integralen Sicherheit werden aktive und passive Sicherheitsfunktionen im Fahrzeug kombiniert, dadurch entstehen neuartige Fahrzeugfunktionen wie Fahrerassistenzsysteme (z. B. Adaptive Cruise Control mit Notbremsfunktion), Pre-Crash-Funktionen oder adaptive Sicherheitssysteme. Deren Funktionsweise und Wirkung passt sich der Art und Schwere der Gefahrensituation an, um Unfälle zu verhindern oder die Unfallschwere zu reduzieren. Dabei führen integrale Sicherheitsfunktionen zu einer immer stärkeren Vernetzung der Systeme im Fahrzeug. Die steigende Komplexität im Entwicklungs- und Integrationsprozess führt zu einer großen Herausforderung, besonders wenn einzelne Systeme nur separat und nicht als Gesamtsystem mit allen Einflussgrößen betrachtet werden.

Ein wesentliches Werkzeug zur disziplinübergreifenden Betrachtung in der Entwicklung neuer Fahrerassistenzsysteme ist die Co-Simulation, bei der unterschiedliche Tools und Simulationsmodelle gekoppelt werden. Damit können disziplinübergreifende Fragestellungen sowie Abhängigkeiten und Interaktionen zwischen einzelnen Systemen einfacher untersucht werden. In dieser Arbeit wird ein Ansatz betrachtet, bei dem mit Hilfe der Co-Simulation der präventive Fußgängerschutz nachgebildet wird. Als Sensor für die Fußgängererkennung werden eine Monokamera und ein kooperativer Sensor verwendet. Die Arbeit befasst sich mit der Implementierung und Integration in das Fahrzeug, Messungen am Fußgängerprüfstand, Erstellung der Modelle und abschließender Anwendung der Simulation bei unterschiedlichen Szenarien.

Der betrachtete Ansatz erlaubt es, den präventiven Fußgängerschutz als Gesamtsystem zu simulieren und Auswirkungen durch Variation einzelner Systemparameter und unterschiedlicher Szenarien zu analysieren. Damit ist es möglich, die Bewertung der Feldeffektivität der Sicherheitsfunktion durchzuführen. Die Parametrisierung der Simulation erfolgt dabei direkt über Ergebnisse von Messfahrten.

Abstract

The continuous development in the field of sensor technology and reliable signal processing is responsible for the substantial progress in the area of integral automotive safety systems. Integral safety systems have the aim to avoid accidents and to reduce the severity of accidents, as well as to further minimize the risk of injury. To achieve this, passive and active safety systems are combined, leading to broad cross-linking of safety functions in the vehicle, like driver assistance systems (e. g. adaptive cruise control with emergency brake), pre-crash systems or adaptive safety systems. As a consequence, the complexity in the development and integration of such distributed safety systems is increasing. This represents a major challenge: currently safety functions are mainly developed and analysed independently instead of considering the entire distributed system. Thus, there is a lack of getting a comprehensive view on the complete system and the influence of interacting parameters is not captured sufficiently.

An essential tool for cross-disciplinary view in the development of new driver assistance systems is co-simulation. Different tools and simulation models of various disciplines are coupled to represent the properties of the complete system. As a result, cross-disciplinary issues can be investigated and dependencies as well as parameter interactions between individual subsystems can be easily analysed. Within this work, a co-simulation-based approach for the validation of a preventive pedestrian protection is demonstrated. A mono camera and a cooperative sensor are used for the pedestrian protection. The master thesis covers (1) the setup of models, (2) the application of the co-simulation with different scenarios and varying parameters, (3) the implementation and integration of the required systems into the car, and (4) the measurements on the test bench and during vehicle test runs.

The proposed comprehensive approach allows to design the preventive pedestrian protection as a complete system in a co-simulation environment based on different scenarios. Further, it supports the evaluation of the field effectiveness of the safety function. The parameterization of the simulation is carried out directly on the results of measurements obtained on a test bench.

EIDESSTATTLICHE ERKLÄRUNG

Ich erkläre an Eides statt, dass ich die vorliegende Arbeit selbstständig verfasst, andere als die angegebenen Quellen/Hilfsmittel nicht benutzt, und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

Graz, am ....

(Markus Schratter)

Inhaltsverzeichnis

Abkürzungsverzeichnis

ADTF	Automotive Data and Time-Triggered Framework
AEB	Autonomous Emergency Braking
ACC	Adaptive Cruise Control
BCS	Boundary Condition Server
CAN	Controller Area Network
dbc	data base CAN
DBF	Digital Beam Forming
DOA	Direction Of Arrival
FIR	Ferninfrarot
GNSS	Global Navigation Satellite System
ICOS	Independent Co-Simulation
Ko-TAG	kooperativer TAG
MicroFW	Micro Framework
NIR	Nahinfrarot
PED	Pedestrian - Fußgänger
pFGS	präventiver Fußgängerschutz
TDMA	Time Division Multiple Access
TOF	Time-of-Flight
TTC	Time-to-Collision
UDP	User Datagram Protocol
UWB	Ultra Wide Band
VEH	Vehicle - Ego-Fahrzeug
WLAN	Wireless Local Area Network
ZGW	Zentrales Gateway

Abbildungsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

In den letzten Jahren führte die Weiterentwicklung der Sensorik im Bereich der Objekterkennung zu wesentlichen Fortschritten in der Integralen Fahrzeugsicherheit. Die grundlegende Zielsetzung der Integralen Sicherheit ist die Vermeidung von Unfällen sowie die Reduktion der Schwere der Unfälle. Um dieses Ziel zu erreichen, werden unterschiedlichste Systeme kombiniert, die eine immer stärkere Vernetzung der Sicherheitsfunktionen im Fahrzeug erfordern, was jedoch auch eine erhöhte Komplexität im Entwicklungs- und Integrationsprozess dieser Sicherheitsfunktionen bedeutet. Die Komplexität stellt bei der Entwicklung von neuen Sicherheitsfunktionen eine große Herausforderung dar, weil mit herkömmlichen Methoden oft nur einzelne Systeme betrachtet werden können und nicht das vernetzte System. Einen umfassenden Überblick über das komplette System zu erhalten, ist mit herkömmlichen Methoden nur bedingt möglich. Die Komplexität macht es notwendig effiziente Methoden zu erarbeiten, mit denen die Entwicklung Integraler Sicherheitssysteme unterstützt werden kann, sowohl im Bezug auf Kosten als auch im Bezug auf die Entwicklungsdauer.
Die Integration der unterschiedlichen Systeme erfordert eine entsprechende disziplinübergreifende Betrachtung. Ein wesentliches Werkzeug stellt die Co-Simulation dar. Dabei werden unterschiedliche Tools und Modelle kombiniert, um die Eigenschaften des vernetzten Gesamtsystems darzustellen. Mit Hilfe der Co-Simulation können disziplinübergreifende Fragestellungen beantwortet werden, sowie Abhängigkeiten und Interaktionen zwischen den einzelnen Teilgebieten aufgezeigt werden.
In dieser Arbeit wird ein Ansatz basierend auf der Co-Simulation betrachtet, mit dem Sicherheitssysteme in einer frühen Phase entwickelt und analysiert werden können. Als Sicherheitssystem wird der präventive Fußgängerschutz herangezogen. Für diese Arbeit werden alle Komponenten modelliert, die bei der Gesamtsimulation benötigt werden. In

der Co-Simulation ist somit das Gesamtsystem inklusive aller relevanten Auswirkungen und Abhängigkeiten nachgebildet und kann dadurch als verteiltes System simuliert werden.
Mit Hilfe von Messfahrten an einem Prüfstand mit definierten Szenarien werden die Modelle parametrisiert und anschließend simuliert. Die Co-Simulation ermöglicht dabei die Durchführung einer Reihe von schnellen und effizienten Simulationen mit verschiedenen externen (z. B. Umgebung) und internen Parametern (z. B. Konfiguration des Funktionsalgorithmus mit unterschiedlichen Strategien). Damit ist es möglich, das Gesamtsystem und die Auswirkung, die durch die Variation einzelner Systemparameter entsteht, zu analysieren.

1.2 Zielsetzung

Am Beispiel des präventiven Fußgängerschutzes soll die Co-Simulationsumgebung ICOS [Vif13] mit dem kooperativen Sensor Ko-TAG [Kof13] und einem kamerabasierten Sensor in einem Versuchsträger erprobt werden. Dazu soll einerseits der Sensor in einen Versuchsträger integriert werden, sodass er ähnlich wie ein kamerabasiertes System auf Fußgänger reagiert und andererseits soll parallel dazu der gesamte Ablauf simuliert werden.
Am Ende soll es möglich sein, unterschiedliche Lastfälle zu simulieren und die aus der Simulation resultierenden Parameter in den Versuchsträger zu übertragen, um anschließend den Lastfall real am Prüfstand abzufahren. Als Anwendungsfall werden die Parameter des Funktionsalgorithmus, der für die Berechnung der Warn- und Bremsstrategie zuständig ist, verwendet. Die dafür nötigen Arbeitsschritte werden in den folgenden Absätzen aufgelistet und kurz beschrieben:

Versuchsträger: Hochrüsten des Versuchsträgers mit dem Ko-TAG-Sensor, der Messtechnik für die Datenaufzeichnung des Ko-TAG-Sensors und der verbauten Serienkamera zur Fußgängererkennung.

Messfahrten: Durchführung von Messfahrten am Fußgängerprüfstand mit einem automatisiert fahrenden Target und Analyse der kamerabasierten Fußgängererkennung hinsichtlich des Warn- und Bremskorridors und der Zeit, bis der Fußgänger als solcher erkannt wird. Ausmessen des Ko-TAG-Sensors am Fußgängerprüfstand hinsichtlich der Reichweite und der Detektionsleistung bei Verdeckung durch ein stehendes Fahrzeug.

Modellierung: Erstellen von Sensormodellen für die Simulation mit der kamera- und Ko-TAG-basierten Fußgängererkennung. Entwicklung eines Funktionsalgorithmus für den präventiven Fußgängerschutz mit dem Ko-TAG-Sensor.

Gesamtsimulation: Integration des Sensormodelles (Ko-TAG oder Kamera) und des Funktionsalgorithmus in die Co-Simulation. Parametrisierung der Simulation anhand der Messfahrten.

Validierung: Überprüfung der Übereinstimmung der Simulationsergebnisse mit den Messfahrten am Fußgängerprüfstand.

Parameterübertragung: Anpassung von Parametern beim Funktionsalgorithmus (z.B. Warnkorridor vergrößern). Simulation mit den neuen Parametern und Übertragung dieser in den Versuchsträger. Durchführung neuerlicher Messungen am Prüfstand.

1.3 Gliederung

In **Kapitel 2** wird der präventive Fußgängerschutz erörtert, dabei werden mögliche Sensoren für die Frontraumüberwachung mit dem Fokus auf die Fußgängererkennung beschrieben. Die Sensoren: Kamera, Radar, Lidar, Nachtsichtsysteme und der kooperative Sensor Ko-TAG werden näher betrachtet. Außerdem wird die Co-Simulation skizziert und Beispiele der Verwendung der Co-Simulation in der Automobilindustrie angeführt. Das gewählte Szenario, anhand dessen der präventive Fußgängerschutz am Prüfstand vermessen und im Nachhinein simuliert werden soll, wird am Ende des Kapitels beschrieben.

In **Kapitel 3** werden die einzelnen Teilmodelle angeführt, die bei der Simulation des präventiven Fußgängerschutzes benötigt werden. Besonders die Teilmodelle, welche im Zuge dieser Arbeit entstanden sind, werden näher beschrieben. Ein weiterer Punkt in diesem Kapitel ist die Gesamtsimulation in der Simulationsumgebung ICOS. Die Einbindung eines Modelles in die Simulationsumgebung ICOS wird anhand der Integration des Ko-TAG-Sensors skizziert.

In **Kapitel 4** wird die Integration des Ko-TAG-basierten und kamerabasierten präventiven Fußgängerschutzes in den Versuchsträger dargestellt. Alle dafür nötigen Umbaumaßnahmen und zusätzlich benötigten Komponenten werden näher erläutert.

Die Durchführung des präventiven Fußgängerschutzes am Fußgängerprüfstand und in der Simulation mit den Sensoren Ko-TAG und Kamera wird in **Kapitel 5** beschrieben. Die durchgeführten Szenarien, die Datenauswertung und die Parametrisierung der Simulation werden dabei näher betrachtet.

Als Abschluss erfolgt in **Kapitel 6** eine Zusammenfassung der Arbeit, im Besonderen werden die aufgetretenen Probleme und Verbesserungsvorschläge diskutiert.

Kapitel 2

Grundlagen

2.1 Sensorik zur Fußgängererkennung

In diesem Kapitel werden mögliche Sensoren für die Fußgängererkennung näher betrachtet, die einerseits bereits in der Serie verwendet werden und andererseits bei zukünftigen präventiven Fußgängerschutzsystemen verwendet werden können.

2.1.1 Radar (Radio Detection and Ranging)

Radarsensoren werden seit 1998 beim Fahrerassistenzsystem aktive Geschwindigkeitsregelung (Adaptive Cruise Control, kurz ACC) eingesetzt. Mittlerweile werden Radarsensoren neben der adaptiven Geschwindigkeitsregelung auch bei aktiven Sicherheitsfunktionen automatischer Notbremsassistent und Spurwechselassistenten eingesetzt, hierfür wird der Sensor für die Abstands-, die Relativgeschwindigkeitsmessung und die Objektdetektion verwendet.

ACC unterstützt den Fahrer bei der Geschwindigkeitsregelung und passt sich dem umgebenden Verkehr an. Sobald im Detektionsbereich des Sensors ein Fahrzeug erkannt wird, erfolgt die Anpassung der Geschwindigkeit auf das vorausfahrende Fahrzeug. Dazu nimmt das System automatisch Brems- und Beschleunigungseingriffe vor, um einen vorgegebenen Abstand zu halten. Beim Notbremsassistenten erfolgt ein automatischer Bremseingriff, wenn eine durch den Radarsensor erkannte Gefahrensituation nicht mehr vermeidbar ist, um die Unfallschwere zu vermindern. Beim Spurwechselassistenten wird durch Radarsenoren im seitlichen Heckbereich des Fahrzeuges der tote Winkel des Fahrers überwacht. Wenn dieser einen Spurwechsel einleitet und ein Fahrzeug befindet sich im toten Winkel bzw. neben ihm, erfolgt eine Warnung des Fahrers, damit dieser auf die Gefahrensituation reagieren kann. [WHW12]

Short Range Radar
Der Short Range Radar (Nahbereichsradar) ist ausgelegt, um Objekte im Nahbereich des Fahrzeuges zu detektieren. Funktionen wie Spurwechselwarnung, Heckaufprallerkennung oder ACC Stop&Go sind damit realisierbar. Hierfür hat der Sensor einen Abstrahlwinkel horizontal von +/- 60° und vertikal von +/- 10°. Der Entfernungsbereich variiert je nach Fahrerassistenzsystem im Bereich von 0,5m und 50m. [Rei10]

Long Range Radar
Für die adaptive Geschwindigkeitsregelung wird ein Long Range Radar (Fernbereichsradar) verwendet, dieser hat im Vergleich zum Nahbereichsradar eine wesentlich größere Reichweite von bis zu 200m. Um dies zu erreichen, beträgt der vertikale Winkel nur +/- 5°. Der horizontale Abstrahlwinkel ist auch wesentlich schmäler, da nur der Bereich vor dem Fahrzeug von Bedeutung ist. [Rei10]

2.1.1.1 Funktionsprinzip

Mit Hilfe von zyklisch gesendeten elektromagnetischen Wellen im Mikro- und Millimeterbereich tastet der Radarsensor den gewünschten Bereich ab. Befindet sich im Sendebereich ein Objekt, erfolgt eine Reflexion am Objekt und ein Teil der Wellen wird vom Sensor wieder empfangen. So kann auf ein Objekt geschlossen werden und aus der Laufzeit der Wellen (Time-of-Flight) kann die Entfernung bestimmt werden. Die Entfernung ergibt sich aus der halben Laufzeit (hin und zurück) und der Ausbreitungsgeschwindigkeit der Wellen. Bei einem Zielabstand von 100 m ergibt sich nur eine Laufzeit von 0,67 μs. Damit die Messung funktioniert, muss beim Empfangen einer Welle auf die gesendete Welle rückgeschlossen werden. Dazu werden für Automobil-Anwendungen die beiden Modulationsarten Pulsmodulation (Pulsradar) oder Frequenzmodulation mit linearen Rampen (FMCW-Radar) verwendet. Die Pulsmodulation funktioniert nach dem Prinzip, dass ein Sinussignal mit einem Puls gefaltet und übertragen wird, dabei verändert sich die Länge des Pulses in Abhängigkeit zu der Zeit. Beim FMCW-Radar wird hingegen die Trägerfrequenz kontinuierlich rampenförmig geändert. [Rei10]
Die Geschwindigkeitsdifferenz zum gemessen Objekt lässt sich mit dem Radarsensor direkt bestimmen, hierfür wird der Dopplereffekt ausgenutzt. Gibt es eine relative Bewegung zwischen dem Sensor und dem zu messenden Objekt, so kommt es zu einer Frequenzverschiebung der reflektierten Trägerfrequenz. Die Geschwindigkeitsdifferenz lässt sich direkt aus der Frequenzdifferenz berechnen. [Rei10]
Neben der Abstands- und Differenzgeschwindigkeitsmessung wird auch noch der Winkel zur genauen Positionsbestimmung benötigt. Hierbei unterscheidet man zwischen den beiden Verfahren: Wellenfrontanalyse und bündelnde Antennensysteme. In den folgenden Ab-

schnitten wird bei den bündelnden Antennensystemen nur auf die am häufigsten verwendeten Verfahren, den Mehrstrahler und das Scanning-Verfahren, eingegangen.

Bündelnde Antennensysteme

Die Radarstrahlen werden nicht als gleichmäßige Kugelwelle gesendet, sondern in eine gewünschte Richtung gebündelt. Dies ist auch notwendig, um eine gewünschte Reichweite zu erreichen, da nur eine begrenzte gesetzliche Sendeleistung erlaubt ist. Durch die Auslegung der Antennencharakteristik erfolgt die erforderliche Verteilung der Sendeleistung in einer Keule mit einem Azimuthwinkel (horizontale Ebene) und einem Elevationswinkel (vertikale Ebene). [WHW12]

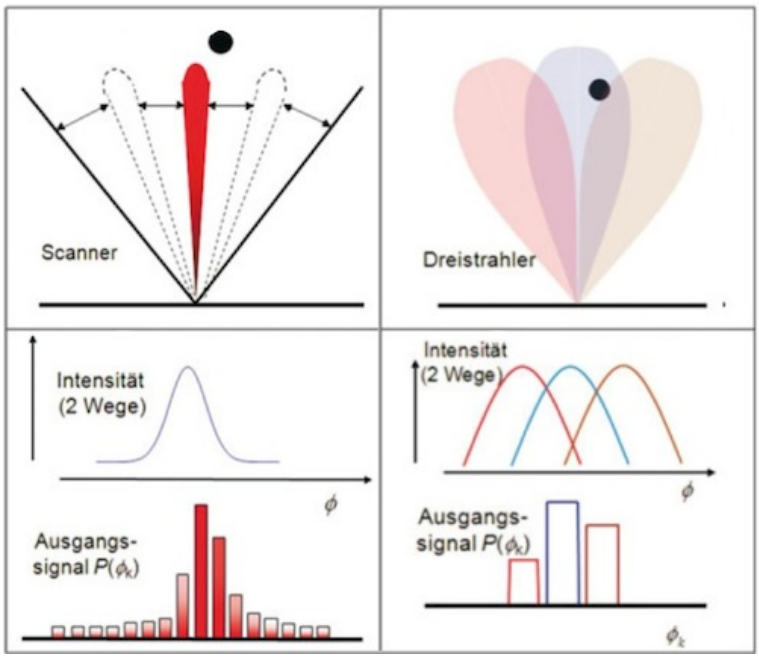

Abbildung 2.1: Bündelnde Antennensysteme - links: Scanning-Verfahren, rechts: Mehrstrahlverfahren [WHW12]

Scanning-Verfahren

Beim Scanning-Verfahren bewegt sich die Antenne mechanisch innerhalb eines Mess- und Auswertezyklus (50-200 ms) über den gesamten horizontalen Erfassungsbereich des Radarsensors. Der hoch gebündelte Radarstrahl hat dabei eine Breite von mindestens 2° und wird in ungefähr 1° Schritten bewegt, wobei aber während eines Messzyklus die Antenne bei den einzelnen Messpunkten nicht stehen bleibt. Damit die Antenne nicht ständig stoppt und beschleunigt, bewegt sich die Antenne kontinuierlich von links nach rechts, wie in Abbildung 2.1 dargestellt ist. Dadurch wird eine Geräuschbildung vermieden und es kommt

zu einem geringeren Energieaufwand. Da sich die Antenne ständig bewegt und zu keiner exakten Winkelposition gemessen wird, erfolgt die Zuweisung des gemessenen Signales zum Mittelpunkt des Messfensters. Aus den einzelnen Messungen und der daraus resultierenden gemessenen Signale kann die Position des reflektierenden Objektes berechnet werden.
Das Scanning hat den Vorteil, dass der Radarstrahl sehr schmal ist (geringe Sendeleistung oder höhere Reichweite) und damit eine hohe Genauigkeit bei der Winkelmessung und eine Trennung von Objekten hinsichtlich des Winkels möglich ist. Dafür ist aufgrund der Schwenkbewegungen am Rand des Sensors keine Messung möglich, da sich der Empfangswinkel gegenüber dem Abstrahlwinkel verkleinert und im Fernbereich kann selbst mit dem schmalen Radarstrahl keine genaue Messung erfolgen. Bei einer Breite des Radarstrahles von 2° und einer Entfernung von 50 m ergibt sich eine ausgedehnte Breite von 1,8 m. [WHW12]

Mehrstrahler
Im Gegensatz zum Scanning-Verfahren bewegt sich beim Mehrstrahler-Verfahren die Antenne nicht, dafür werden aber mindestens zwei Antennen mit unterschiedlicher Strahlungscharakteristik verwendet. In Abbildung 2.1 ist das Prinzip des Mehrstrahlers dargestellt, die Auswertung der Position des Objektes erfolgt über die Zusammenfassung der Amplituden der einzelnen Antennen. [WHW12]

Wellenfrontanalyse
Die Wellenfrontanalyse ist auch unter den beiden Begriffen Digital Beam-Forming (DBF) und Direction of Arrival (DOA) bekannt. Bei diesem Verfahren wird mit Hilfe von mehreren Antennen der Einfallswinkel der reflektierten Welle des Objektes ausgewertet. Durch mehrere linear zueinander angeordnete Antennen kann die Phasenverschiebung einer auftreffenden Welle gemessen und daraus der Winkel zum Objekt berechnet werden.
Beim Verfahren nimmt man an, dass die reflektierte Welle vom Objekt als ebene Welle beim Sensor auftrifft. Befindet sich ein Objekt genau vor dem Sensor, so empfangen alle Antennen zur selben Zeit dieselbe Welle, keine Phasendifferenz ist messbar. Befindet sich jedoch ein Objekt z. B. rechts vom Sensor, so empfängt die rechte Antenne die reflektierte Welle kurz vor der mittleren Antenne, alle Empfangssignale haben eine unterschiedliche Phase. Durch den Phasenunterschied lässt sich der Laufzeitunterschied zwischen den einzelnen Antennen bestimmen und daraus lässt sich auf den Winkel zum Objekt rückschließen. [WHW12]

Mit zwei Antennen kann theoretisch der Winkel zu einem Objekt bestimmt werden. Allgemein lässt sich sagen, dass zur Bestimmung von n Objekten n+1 Antennen benötigt werden. In der Automobilindustrie kommt derzeit hauptsächlich die Winkelmessung mit

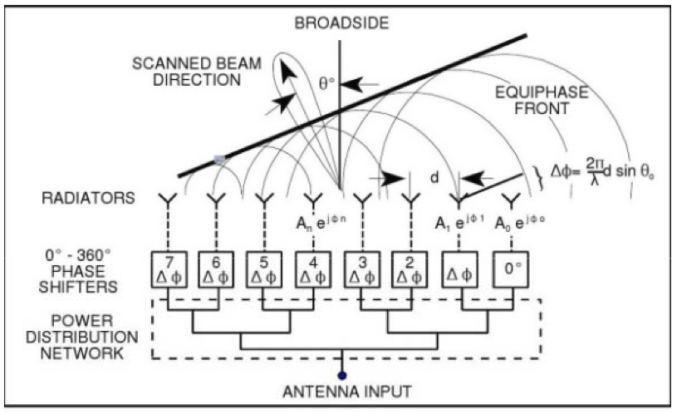

Abbildung 2.2: Schematische Darstellung der Wellenfrontanalyse [WHW12]

hoch bündelnden Antennen zum Einsatz. Die Wellenfrontanalyse verursacht einen wesentlich höheren Hardware- als auch Software-Aufwand, jedoch steckt in dieser Technologie zukünftig noch großes Potential. [Rei10], [WHW12]

2.1.1.2 Frequenzspektrum

Radarsensoren müssen in der Automobilindustrie bestimmte Anforderungen erfüllen. Einerseits steht nur ein begrenzter Bauraum zur Verfügung, gesetzlich vorgegebene Frequenzbänder müssen verwendet werden und eine massentaugliche Produktion der Sensoren muss gegeben sein. Andererseits soll aber der Sensors eine hohe Qualität von Sicherheitsfunktionen durch gute Ortungsqualität liefern und dadurch eine entsprechende Kundenakzeptanz für diese Systeme garantieren. [WHW12]
Bei Sensoren mit einer Frequenz von unter 20 GHz tritt das Problem der Integration auf, da aufgrund der benötigten Antennengröße keine ausreichende Strahlenbündelung bzw. Begrenzung des Erfassungsbereiches mit dem zur Verfügung stehenden Bauraum möglich ist. Hingegen haben Sensoren mit Frequenzen über 100 GHz das Problem der atmosphärischen Dämpfung und sie sind mit der heutigen Technik noch nicht für eine massentaugliche Produktion geeignet. Aus diesen Gründen werden Radarsensoren im Nahbereich derzeit hauptsächlich im 24 GHz-UWB (ultra-wide-band) und im Fernbereich bei 77 GHz betrieben. Das 24 GHz-UWB ist nur bis Juni 2013 in der Europäischen Union zugelassen, neu zugelassene Sensoren müssen danach im 79 GHz Frequenzband arbeiten, dieses reicht von 77-81 GHZ und beinhaltet auch das bestehende 77 GHz Band. Mit diesem Frequenzband können alle derzeit relevanten Sicherheitsanforderungen erfüllt werden. [Rei10], [WHW12]

2.1.1.3 Anwendung Fußgängererkennung

Zur Fußgängererkennung ist Radar aufgrund der schwachen Reflexion des Fußgängers nur bedingt geeignet. In Kombination mit einer Mono- oder Stereokamera kann Radar jedoch zu einer verbesserten Fußgängererkennung verwendet werden. Die Kamera ist sehr präzise bei der Erkennung von Bewegungsmustern und der Identifizierung von Objekten. Mit Hilfe der Kamera können Fahrbahnmarkierungen, wie z. B. die Begrenzungslinien, erkannt und dadurch die Gesamtsituation besser eingeschätzt werden. Falschauslösungen können somit vermieden werden. Die Sensordatenfusion nutzt eine Redundanz der Objekterkennung und eine präzisere Positionsbestimmung des Fußgängers. Die Vorteile beider Sensoren ergänzen sich und über zwei unterschiedliche physikalische Methoden erfolgt die Objekterkennung. Damit können Falschauslösungen ausgeschlossen werden und auch eine stärkere Verzögerung des Fahrzeuges ist möglich. Aufgrund zukünftiger Anforderungen des EuroNCAP beim präventiven Fußgängerschutz, ist eine vermehrte Verwendung der Sensordatenfusion mit Kamera und Radar zu erwarten. [Hei12]

2.1.2 Lidar (Light Detection and Ranging)

Lidarsensoren funktionieren nach einem ähnlichen Prinzip wie Radarsensoren, aus diesem Grund wird nur kurz auf den Lidarsensor eingegangen. Der Hauptunterschied liegt im Frequenzspektrum. Anstatt Mikrowellen werden beim Lidarsensor Ultraviolettstrahlen, Infrarotstrahlen oder Strahlen aus dem sichtbaren Bereich des Lichtes verwendet. [WHW12]

2.1.2.1 Funktionsprinzip

Die Abstandsmessung erfolgt wie beim Radar über eine Laufzeitmessung. Dafür sendet ein Laser einen Lichtpuls aus, dieser wird an einem Objekt reflektiert und über eine Photodiode empfangen.

Der Abstand zum reflektierenden Objekt ist direkt proportional zur Zeit des gesendeten Lichtpulses, die Berechnung erfolgt nach folgender Formel:

$$
\begin{aligned}
&r = \frac{c_0 * t_{of}}{2} \\
&r ... Abstand[m] \\
&c_0 ... Lichtgeschwindigkeit[300000\frac{m}{s}] \\
&t_{of} ... Laufzeit[s]
\end{aligned}
\tag{2.1}
$$

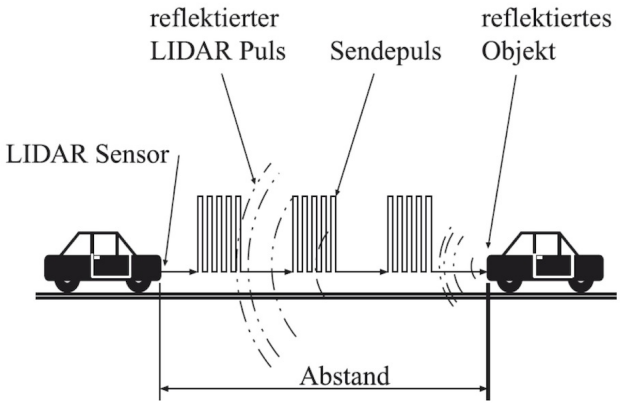

Abbildung 2.3: Lidar - Laufzeitmessung (Time-of-Flight), [WHW12]

Die Reichweite von Lidarsensoren ist technisch begrenzt durch die Intensität des ausgehenden Lichtpulses und durch die Empfindlichkeit der Empfangsdiode. Aufgrund des nicht funktionierenden natürlichen Schutzmechanismus des menschlichen Auges gegenüber Laserstrahlen (Schließen der Pupillen außerhalb des sichtbaren Spektrums) kann die eingebrachte Energie des Strahles im schlimmsten Fall zu einer Zerstörung der Netzhaut führen. Somit ist nur eine gesetzlich begrenzte Ausgangsspitzenleistung (max. 70 W) und Ausgangsdurchschnittsleistung (max. 5 mW) erlaubt. Deshalb verwendet man zur Umgebungserfassung nur gebündelte Strahlen. Damit dennoch das gesamte Umfeld vor dem Fahrzeug erfasst wird, werden dafür Mehrstrahlsysteme oder schwenkende Ein-/Mehrstrahler verwendet, wie in Abbildung 2.4 dargestellt ist. Die Winkelmessung für die Bestimmung des lateralen Abstandes und eine mehrfache Objekterkennung ist dadurch möglich. [WHW12]

2.1.2.2 Anwendung Fußgängererkennung

ACC-Systeme sind bereits seit mehreren Jahren am Markt und erste AEB-Systeme, in denen ein Lidarsensor für die Objekterkennung verwendet wird, sind auch seit kurzem am Markt verfügbar. Der Vorteil von Lidar gegenüber Radar ist die Genauigkeit bei der Winkelmessung, besonders bei scannenden Lidars wird eine Messgenauigkeit im cm-Bereich erzielt, wodurch sie sich von heute verfügbaren Radarsensoren deutlich unterscheiden. [Rei10] Ein weiterer Vorteil ist das breitere Einsatzgebiet gegenüber Radar, da eine homogenere Reflexion der Lichtstrahlen an Objekten erfolgt, Personen können somit sehr gut detektiert werden. Jedoch liefern Lidarsensoren wie Kameras bei schlechten Witterungsverhältnissen keine brauchbaren Ergebnisse. [WHW12]

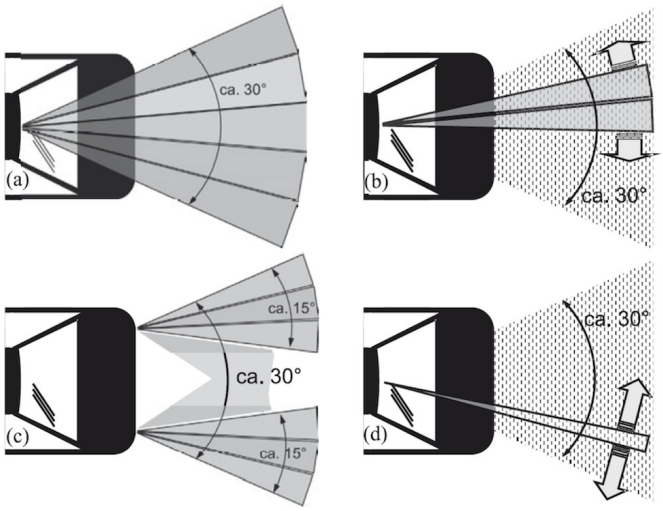

Abbildung 2.4: Mögliche Strahlsensorik Lidar - (a) Multibeam starr, (b) Multibeam SWEEP. (c) Multibeam verteilt, (d) Singlebeam SCAN, [WHW12]

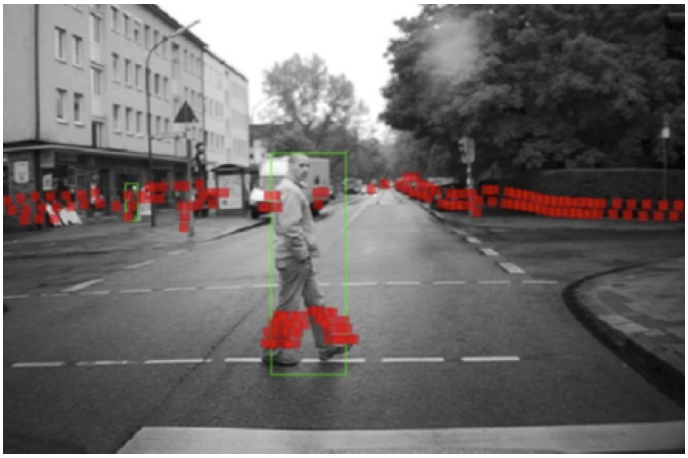

Abbildung 2.5: Lidar - Messaufzeichnung mit Objekterkennung [BMW10]

2.1.3 Nachtsichtsysteme

Laut [Rei10] passieren etwa 40 % aller tödlichen Unfälle bei Nacht, obwohl zu dieser Zeit nur 20 % der Fahrten erfolgen. Die Ursache hierfür sind meist schlechte Sichtverhältnisse und Blendungen durch den Gegenverkehr.
Abhilfe zur Erkennung von Gefahrensituationen mit Fußgängern oder Tieren neben bzw. auf der Fahrbahn schaffen Nachtsichtsysteme. Sie haben eine Reichweite von bis zu 300 m. In diesen Entfernungen sind Objekte bei Dunkelheit für den Fahrer mit freiem Auge noch nicht sichtbar.
Bei Nachtsichtsystemen unterscheidet man zwischen Nah- und Fern-Infrarot-Systemen, in

Abbildung 2.6 ist der schematische Unterschied zwischen den beiden Systemen dargestellt. [Rei10]

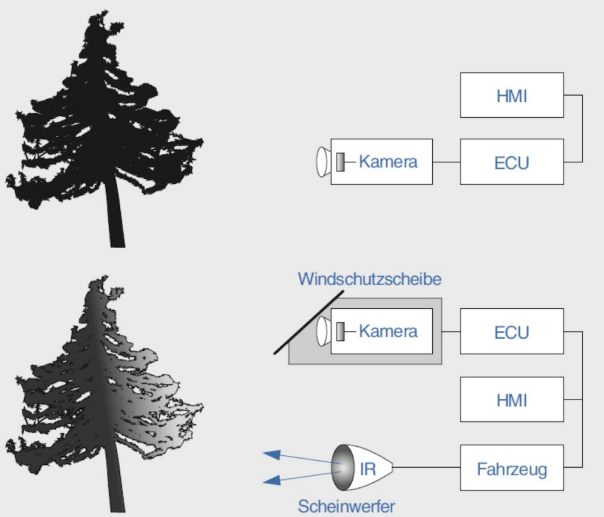

Abbildung 2.6: Schematischer Aufbau Nachtsichtsysteme - oben: Fern-Infrarot-System, unten: Nah-Infrarot-System [Rei10]

2.1.3.1 Nah-Infrarot-Systeme (NIR)

Für das menschliche Auge nicht sichtbare Infrarotstrahlung zwischen 800 nm und 1000 nm wird bei Nah-Infrarot-Systemen verwendet. Die Wellenlänge liegt nahe dem sichtbaren Lichtspektrum, daher kommt auch der Name „Nah-Infrarot“. Nah-Infrarot-Systeme sind aktive Sensoren, sie leuchten mit Infrarot-Scheinwerfern den Bereich vor dem Fahrzeug aus, wie in Abbildung 2.7 dargestellt. Die Infrarotstrahlung wird an Gegenständen reflektiert und von einer Videokamera z. B. an der Windschutzscheibe aufgenommen. Realisiert werden Nah-Infrarot-Systeme durch zusätzliche Scheinwerfer in den Frontscheinwerfern, diese bestehen aus Halogenlampen mit einem optischen Filter, der nur die gewünschte Infrarot-Strahlung durchlässt. Zusätzlich zum Infrarotbild wird auch das Videobild aufgezeichnet und kann so für weitere Funktionen genutzt werden. [Rei10]

2.1.3.2 Fern-Infrarot-Systeme (FIR)

Fern-Infrarot-Systeme sind rein passive Sensoren, sie werten nur die abgestrahlte Wärmestrahlung von Gegenständen aus. Im Gegensatz zu NIR-Systemen wird keine zusätzliche Infrarotquelle zur Beleuchtung der Umgebung benötigt. Die empfangene Wellenlänge der

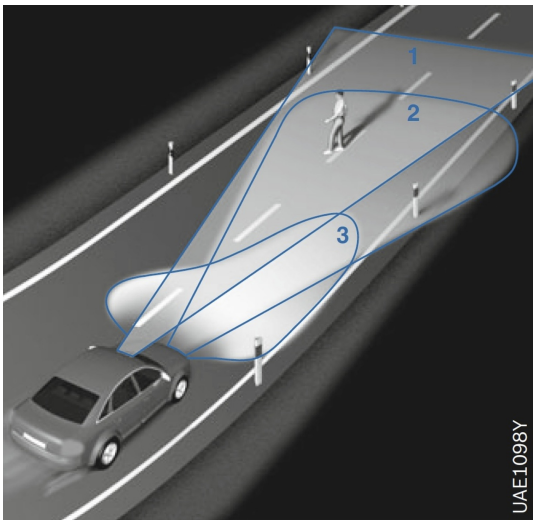

Abbildung 2.7: Ausleuchtung Nah-Infrarot-System - 1: Erfassungsbereich Videosensor, 2: Infrarotkegel, 3: Abblendlicht [Rei10]

Wärmestrahlung liegt im Bereich zwischen 7 und 12 μm, daraus berechnet der Sensor das Wärmebild des Bereiches vor dem Fahrzeug.

Wärmebildkameras können nicht hinter der Windschutzscheibe montiert werden, da der benötigte Wellenlängenbereich vom Windschutzscheibenglas gefiltert wird. Abhilfe schafft nur eine Platzierung der Wärmebildkamera am Außenbereich des Fahrzeuges hinter einer Siliziumscheibe. [Rei10]

2.1.3.3 Anwendung Fußgängererkennung

Neben der graphischen Darstellung des Nachtsichtbildes z. B. im Kombiinstrument oder am Multimedia-Bildschirm verfügen derzeitig am Markt erhältliche Systeme über eine Fußgängererkennung. Dadurch wird der Fahrer nur bei Gefahrensituationen gewarnt, er muss nicht ständig auf das Wärmebild schauen und kann sich somit vollständig auf die Straße konzentrieren.
Die Objekterkennung und -identifizierung erfolgt nach demselben Prinzip wie bei kamerabasierten Fußgängererkennungssystemen. Die Bilddaten werden von einem Steuergerät ausgewertet und mit den aktuellen Fahrzeugdaten (Geschwindigkeit, Lenkwinkel, Warnbereich, ...) wird die Gefahrensituation analysiert. Befindet sich ein Fußgänger auf Kollisionskurs, erfolgt eine Warnung des Fahrers und der Fußgänger wird bei manchen Systemen durch Lichtspots gezielt angestrahlt.
Vergleicht man FIR und NIR-Systeme für die Anwendung zur Fußgängererkennung, so

liegt der Vorteil systembedingt bei Fern-Infrarot-Systemen, da diese bis zu 300 m weit sehen, ungefähr doppelt so weit wie NIR-Systeme. Außerdem werten FIR-Systeme nur die Wärme der Objekte aus. Blendungen von anderen Lichtquellen, wie Ampeln, Scheinwerfern oder ebenen Oberflächen werden vom Sensor nicht wahrgenommen, da diese nicht im Frequenzbereich des Sensors liegen. Als Seriensystem ist dieser aktuell aufgrund der hohen Kosten nicht geeignet und nur als Sonderausstattung verfügbar. [Rei10]

2.1.4 Kooperative Sensorik am Beispiel von Ko-TAG

Neben der klassischen Sensorik (Kamera, Lidar, Radar) ist es auch möglich andere Verkehrsteilnehmer mit kooperativer Sensorik zu identifizieren. Mit Hilfe von kooperativer Sensorik ist es möglich, Sichtverdeckungen zu überwinden und bereits wesentlich früher eine Gefahrensituation zu erkennen. Eine exakte relative Positionsbestimmung des Fußgängers ist möglich, bevor dieser überhaupt optisch sichtbar ist. Zusätzlich ist neben der Positionsbestimmung auch noch ein gleichzeitiger Datenaustausch möglich. Daten, die in der Regel durch lokale Sensoren nicht direkt erfasst werden können, wie z. B. die Geschwindigkeit des Verkehrsteilnehmers, können übertragen und ausgewertet werden. Die Positionsbestimmung der Verkehrsteilnehmer kann einerseits durch globale Navigationssatellitensysteme (GNSS) erfolgen oder durch Sensorik im Fahrzeug, mit der die Position des Verkehrsteilnehmers erfasst wird. GNSS haben den Nachteil, dass die Positionsbestimmung von der Qualität und Verfügbarkeit der satellitenbasierten Positionsbestimmung abhängig ist und besonders im städtischen Bereich eine Genauigkeit von > 1 m erreicht wird. Mit der verbauten Sensorik im Fahrzeug kann die Position mit einer Genauigkeit < 1 m realisiert werden.

Im Forschungsprojekt Ko-Tag wurde ein kooperativer Sensor entwickelt, mit dem eine exakte Positionsbestimmung auch bei Sichtverdeckung möglich ist und der den Car-2-X Kommunikationsstandard 802.11p im 5,9 GHz Frequenzband verwendet. Beim Ko-TAG-System trägt der Fußgänger einen Transponder, der vom Fahrzeug eindeutig als solcher lokalisiert und identifiziert werden kann. [Sch12]

2.1.4.1 Forschungsprojekt Ko-TAG

Basierend auf den Forschungsprojekten AMULETT [Amu09] und Watch-Over [Wat08], die sich mit Ansätzen zur Transponderortung befassten, wurde Ko-TAG weiterentwickelt. Ko-TAG ist der Forschungsinitiative „Kooperative Fahrzeugsicherheit (Ko-FAS)“ [Kof13] angesiedelt und wurde teilweise durch Mittel des deutschen Bundesministeriums für Wirtschaft und Technologie gefördert. Das Ziel ist durch die Verbundprojekte Ko-TAG, Ko-PER

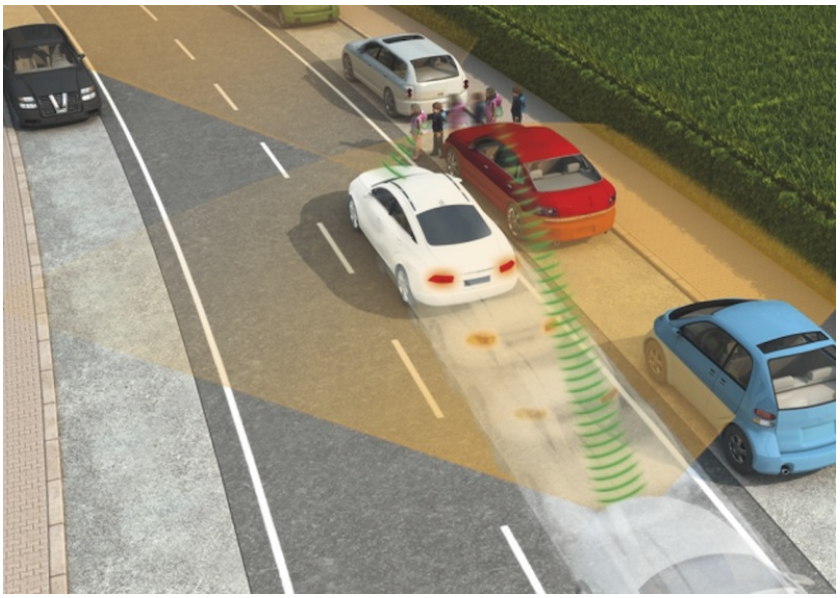

Abbildung 2.8: Ortung und Klassifikation von verdeckten Fußgängern durch Ortung mitgeführter Transponder [Sch12]

und Ko-KOMP eine Steigerung der Verkehrssicherheit zu erreichen. Dies wird versucht, indem dem Verkehrsteilnehmer kritische Verkehrssituationen bereits frühzeitig dargestellt werden. [Sch12]

2.1.4.2 Systemübersicht

Ko-TAG besteht aus zwei über Funk kommunizierenden Komponenten: der Ortungseinheit im Fahrzeug und dem Transponder am Fußgänger. Über den Car-2-X Kommunikationsstandard 802.11p erfolgt die Kommunikation zwischen den beiden Teilnehmern. Die Positionsbestimmung des Transponders erfolgt über ein Laufzeit- und ein Phasenmessverfahren, bei denen die Teilnehmer nicht zeitsynchronisiert sein müssen. Das System verwendet das Prinzip des Sekundärradars. Bei diesem Verfahren antwortet der zu bestimmende Transponder auf die Positionsabfrage der Ortungseinheit, diese wertet die Antwort aus und berechnet daraus die aktuelle Position. [Sch12]

- **Transponder**
  Neben der Einheit für die Kommunikation mit dem Fahrzeug ist beim Transponder auch eine Inertialsensorik integriert, dadurch lassen sich die Beschleunigungen und Drehraten messen. Mit dieser zusätzlich übertragenen Information zum Fahrzeug lässt sich der Bewegungsablauf des Fußgängers wesentlich schneller und genauer vorhersagen, als nur durch eine Ortung vom Fahrzeug aus möglich wäre.

Bei abrupten Geschwindigkeits- oder Richtungsänderungen kann diese Zusatzinformation in nahezu Echtzeit ausgewertet werden. Besonders beim präventiven Fußgängerschutz ist dies von großem Vorteil, einerseits können Falschauslösungen vermieden werden und andererseits können plötzlich auftretende Gefahrensituationen durch eine frühe Systemreaktion entschärft werden, z. B. wenn ein Fußgänger aus dem Stand Richtung Fahrschlauch losrennt. Die Inertialsensorik bietet zusätzlich den Vorteil, dass aufgrund des charakteristischen Beschleunigungsmusters beim Gehen festgestellt werden kann, ob der Fußgänger geht oder sich in einem Fahrzeug befindet und damit nicht relevant ist.

- **Ortungseinheit**
  Die Ortungseinheit im Fahrzeug besteht aus einem Antennen-Array und einer Sende- und Empfangseinheit für die Kommunikation mit den Transpondern. Für die Steuerung des Sensors sowie die Bestimmung und Weiterleitung der Rohdaten wird ein Controller verwendet. Für die Auswertung der Rohdaten und der weiterführenden Berechnung der einzelnen Trajektorien wird ein Computer verwendet.
  Um eine optimale Sende- und Empfangsleistung der Ortungseinheit zu erreichen, sollten sich die Antennen im vorderen Bereich des Fahrzeuges befinden. Um aber nicht die Fahrzeugsicherheit zu beeinflussen, ist die Ortungseinheit bei den Versuchsfahrzeugen direkt hinter dem vorderen Stoßfänger platziert.

2.1.4.3 Positionsbestimmung des Transponders

Entfernungsmessung

Zur Bestimmung der Entfernung zwischen Transponder und Ortungseinheit wird das Time-of-Flight-Verfahren verwendet. Mit dem Versenden eines Data Burst mit der Länge T_s startet die Ortungseinheit einen Messzyklus. Alle Transponder, die sich in der Reichweite befinden und das Signal empfangen, antworten mit einer fixen oder einer vielfachen Wartezeit n*T_w (TDMA – Time-Division-Multiple-Access). Durch die individuellen Wartezeiten der einzelnen Transponder ist es möglich, diese voneinander zu unterscheiden. Der Zeitschlitz wird bei der Registrierung des Transponders bei der Ortungseinheit über ein Protokoll ausgehandelt. [RSMB09]

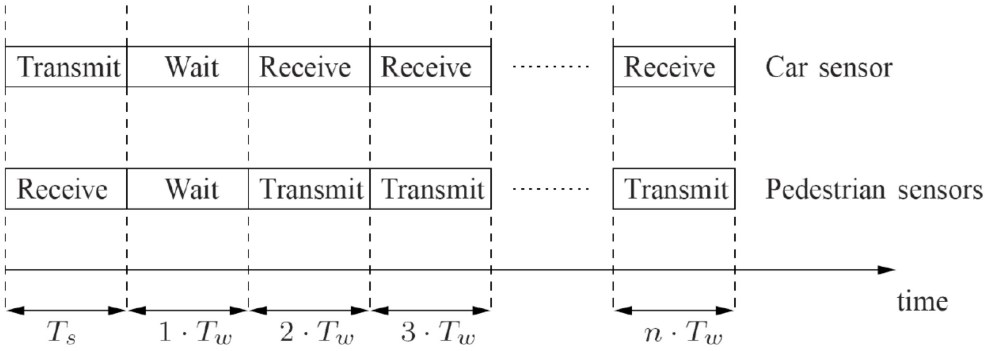

Abbildung 2.9: Zeitschlitze eines Messzyklus [RSMB09]

Der Transponder lauscht ständig nach einer bekannten Folge von Pseudozufallsbits. Nach dem Empfang wartet der Transponder eine definierte Zeit und antwortet der Ortungseinheit. Diese führt ständig eine Korrelation zwischen dem empfangenen und gesendeten Signal durch, damit kann die Empfangszeit exakt bestimmt werden und durch Abzug der Sendezeit ergibt sich die Gesamtlaufzeit. Die Entfernung zwischen der Ortungseinheit und dem Transponder berechnet sich nach folgender Formel:

$$\Delta s = \frac{T_{gesamt} - T_w}{2} * c_0 \tag{2.2}$$

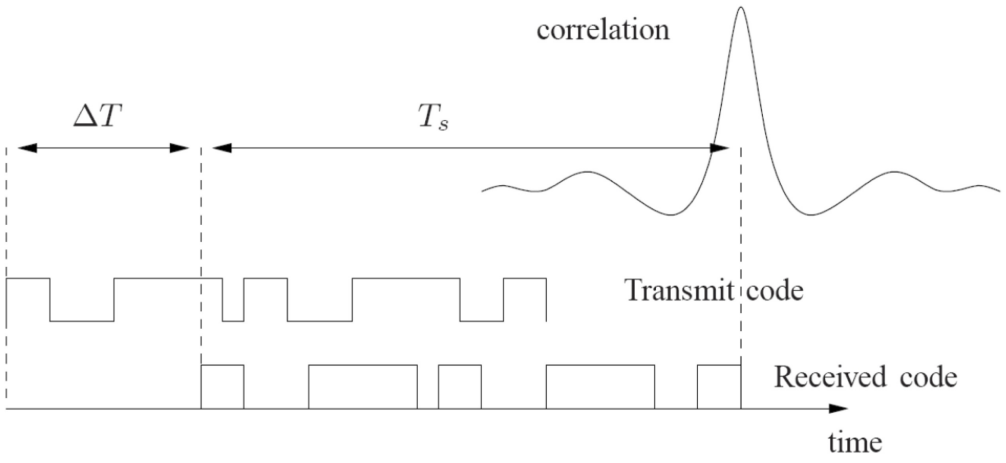

Abbildung 2.10: Schematische Darstellung des Signalablaufs zwischen Ortungseinheit und Transponder [RSMB09]

Aufgrund von Reflexionen an Hindernissen kommt es zu Mehrwegausausbreitung und dadurch zu mehreren Abstandshypothesen eines Transponders. Mit Hilfe von hochauflösenden Schätzalgorithmen ist es möglich die unterschiedlichen Signallaufzeiten zu bestimmen. Im Idealfall handelt es sich bei den, kürzesten gemessenen Abstand um den direkten Pfad zum Transponder, jedoch sind auch Fälle möglich, bei denen es zu Wellenauslöschungen kommt und dieser nicht bestimmt werden kann. [RSMB09]

Winkelmessung
Mit einem Mehrfachantennensystem wird der Winkel zum Transponder nach dem DoA-Verfahren (Direction-of-Arrival) gemessen. Durch die Phasenverschiebung des empfangenen Signals zwischen den einzelnen Antennen kann der Winkel zum Transponder berechnet werden. Die Messung des Winkels erfolgt nach demselben Prinzip wie bei Radarsensoren mit mehreren Empfangsantennen. Im Kapitel 2.1.1.1 wird näher auf das DoA-Verfahren eingegangen.
Wie bei der Entfernungsmessung treten auch bei der Winkelmessung Mehrwegeausbreitungen auf, dadurch kommt es zu mehreren Winkelhypothesen. Mit Hilfe der Abstandsmessung und durch die Anwendung eines Multi-Hypothesen-Tracking werden nicht plausible Hypothesen verworfen. [RSMB09]

2.1.4.4 Protokoll

Für die Kommunikation zwischen Ortungseinheit und Transponder sind drei Kanäle vorgesehen, diese sind zeit- und frequenzmoduliert und für folgende Funktionen zuständig:

- **Management-Kanal**
  In diesem Kanal registrieren sich die Transponder bei der Ortungseinheit und bekommen den Ablauf in den weiteren Kanälen zugewiesen (z. B. Wartezeit T_W nach Anfrage). Gegenseitige Störungen durch mehrere Ortungseinheiten können somit verhindert werden, da immer nur ein Ko-TAG-Sensor aktiv ist und Daten sendet. [Sch12]

- **Entfernungsmessung (ToF)**
  Über das Time-of-Flight Verfahren wird die Entfernung bestimmt. Dabei ist zu beachten: Je größer die Bandbreite des Kanals ist, desto genauer kann die Distanz zum Transponder aufgelöst werden. Um den Durchsatz der Entfernungsmessungen zu erhöhen, wird jedem Transponder ein Timeslot mit der Wartezeit T_w für ein TDMA Verfahren zugeordnet. [Sch12]

- **Winkelmessung (DoA)**
  Die Winkelmessung verhält sich zur Entfernungsmessung genau umgekehrt, je geringer die Bandbreite ist, desto genauer ist die Winkelmessung. Für die Winkelmessung

wird kein eigener Kanal verwendet, sie erfolgt durch Kommunikationspakete zwischen Transponder und Ortungseinheit. [Sch12]

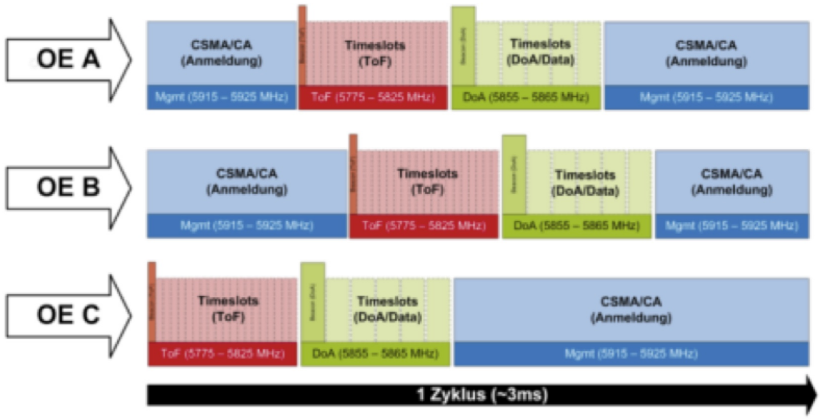

Abbildung 2.11: Protokoll zur Datenübertragung und Ortung [Sch12]

2.1.4.5 Synergien mit Car-2-X-Kommunikation

Ko-TAG beruht auf bestehenden Standards im Bereich der Car-2-X-Kommunikation. Für die Kommunikation zwischen den einzelnen Teilnehmern wird der IEEE 802.11p Standard im 5,9 GHz Frequenzband verwendet. Dies ermöglicht eine Erleichterung bei der Standardisierung und Industrialisierung. Für die Winkelmessung sowie für den Datenaustausch zwischen der Ortungseinheit und den Teilnehmern kann auf den bestehenden Standard zurückgegriffen werden. Lediglich für die Abstandsmessung ist ein zusätzliches Frequenzband notwendig, in dem über ein TDMA-Verfahren die sequentielle Abstandsmessung der einzelnen Teilnehmer erfolgt. [Sch12]

2.1.4.6 Anwendung Fußgängererkennung

Kooperative Sensorik bietet den Vorteil in wenigen Mikrosekunden die Position eines Verkehrsteilnehmers zu bestimmen, auch wenn dieser noch nicht sichtbar ist. Bei Fußgängern und Radfahrer bietet dies entscheidende Vorteile, besonders bei Sichtverdeckung kann, in einer kritischen Situation, eine frühere Systemreaktion eingeleitet werden. Aufgrund der zusätzlich übertragenen Information (Geschwindigkeit und Beschleunigung) durch den Transponder kann die Situation im Fahrzeug wesentlich besser eingeschätzt werden, die Wirksamkeit des Systems lässt sich damit steigern.
Zusätzlich bietet Ko-TAG die Möglichkeit, die Anforderungen an die Security bei der Kommunikation zu verringern, da durch die Entfernungs- und Winkelmessung die Position des

Transponders aktiv gemessen wird. Bei anderen Systemen, die die absolute Position übertragen, wird nur diese ausgewertet und nicht überprüft, ob die Datenpakete von dieser Position kommen. Somit haben die Daten bei Ko-TAG bereits eine hohe Zuverlässigkeit und müssen nicht durch komplexe Security-Mechanismen abgesichert werden. Dies führt zu einer effizienteren Bandbreitennutzung, da der Overhead bei den Datenpaketen verringert werden kann.
Eine Integration des Transponders in mobile Endgeräte (z. B. Smartphone) würde zu einer raschen Verbreitung führen und zu einer Erhöhung der Wirksamkeit beitragen. Um eine Kundenakzeptanz zu erreichen, darf es dabei aber zu keinen großen Einschränkungen, z. B. durch zusätzliches Gewicht oder einen stark erhöhten Energieverbrauch, beim Benutzer kommen. Die Technologie ist nicht nur auf Fußgänger beschränkt, auch andere Verkehrsteilnehmer wie Radfahrer aber auch Fahrzeuge können mit dem System ausgerüstet werden, um eine präzisere Ortung zu ermöglich. [Sch12]

2.1.5 Kamerabasierte Fußgängererkennung

Die Hauptaufgabe von Kamerasystemen in Fahrzeugen ist nicht die Aufnahme einer Szene, sondern die Interpretation und Wahrnehmung von relevanten Szenarien. Mit einer Kamera ist es möglich, mehrere Fahrerassistenzsysteme zu realisieren. Aus dem Kamerabild können die Fahrbahnlinien für den Spurhalteassistenten, Verkehrsschilder und Objekte (Personen, Fahrzeuge usw.) zur Situationsanalyse gewonnen werden. [Fen10]

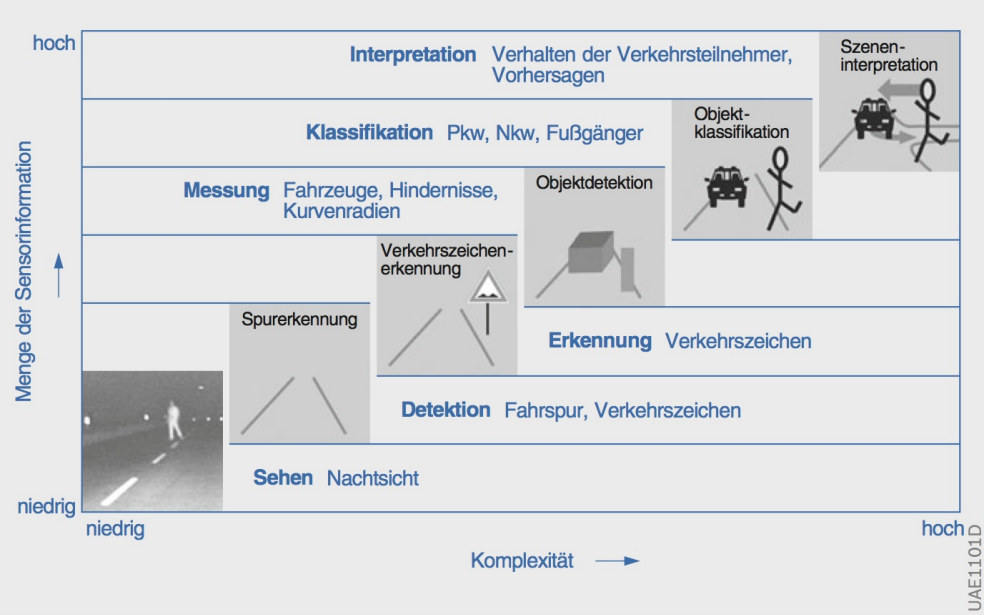

Abbildung 2.12: Stufen der Bildverarbeitung in Abhängigkeit zur Komplexität [Rei12]

Die Anforderungen der einzelnen kamerabasierten Assistenzsysteme in Abhängigkeit zur Komplexität bei der Bildverarbeitung und zur benötigten Sensorinformation ist in Abbildung 2.12 dargestellt. Der präventive Fußgängerschutz benötigt dabei aufgrund der Szeneninterpretation die höchsten Anforderungen.

2.1.5.1 Funktionsprinzip

Der Ablauf bei einem kamerabasierten System, von der Aufnahme durch den Bildsensor bis zur Auslösung einer Aktion durch die Assistenzfunktion, ist in Abbildung 2.13 illustriert. Bei einer Kamera wird das empfange Licht mit Hilfe eines lichtempfindlichen Bildsensors in ein digitales Signal umgewandelt, um in weiterer Folge das aufgenommene Bild digital auszuwerten. Die Bildqualität ist dabei abhängig von der Sensorgröße und der Anzahl der Bildpunkte. Eine hohe Anzahl an Bildpunkten führt dabei nicht automatisch zu einem besseren Bild, da die einzelnen Bildpunkte kleiner sein müssen, um auf den Sensor Platz zu finden. Dadurch sinkt die Lichtempfindlichkeit, jedoch kann dafür eine genauere Klassifizierung der Objekte stattfinden, zum Nachteil einer schlechteren Empfindlichkeit bei dunkleren Sichtbedingungen. Aufgrund einer besseren Bildqualität bei Gegenlicht und Helligkeitswechsel (z. B. bei Tunnelausfahrten) wird in der automobilen Anwendung hauptsächlich der CMOS-Bildsensor verwendet. [Fen10]

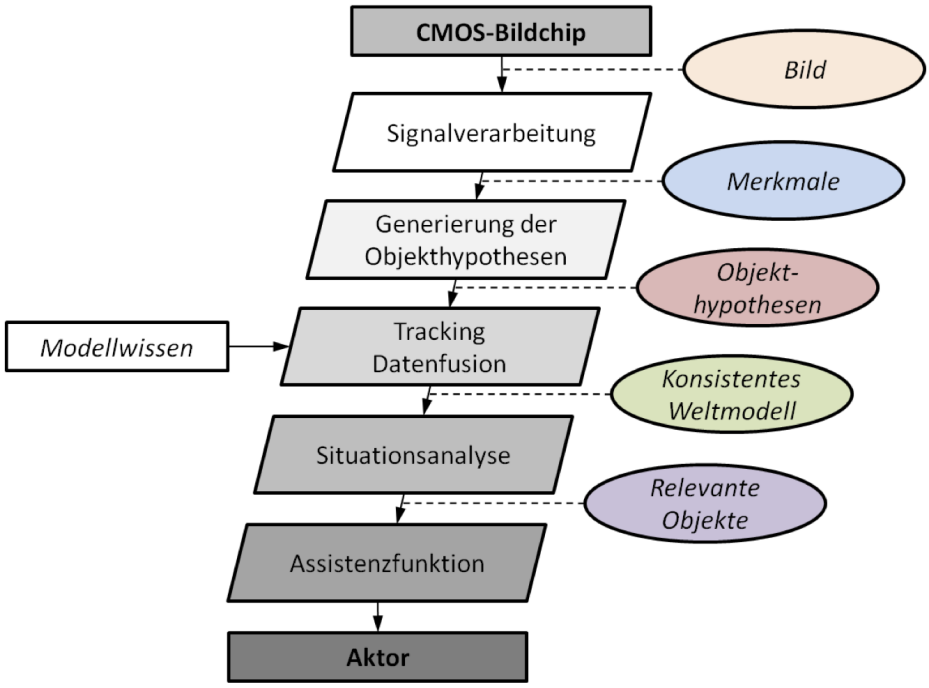

Abbildung 2.13: Kamerabasierte Fahrerassistenzsystem - Vom Bild zur Aktion [Rei12]

Nach der Aufzeichnung des Bildes wird eine Signalverarbeitung durchgeführt. Hierbei wird eine Kantendetektion angewendet und nach Mustern gesucht, um relevante Merkmale aus dem Bild zu erhalten. Aus diesen gewonnenen Merkmalen können Objekthypothesen generiert werden. Aus zusammenlaufenden Linien im unteren Bereich des Bildes kann z. B. auf die Fahrspuren rückgeschlossen werden, runde Kreise stellen Verkehrszeichen dar oder eine Rechteck im Bild stellt ein Fahrzeug dar. Im nächsten Schritt erfolgt mit Hilfe vom Modellwissen und den Objekthypothesen eine Datenfusion und ein Tracking. Daraus entsteht ein Weltmodell mit allen relevanten Daten wie z. B. dem Fahrschlauch des Fahrzeuges, vorausfahrenden Fahrzeugen oder Fußgänger im Bildbereich. Aus dieser aktuellen Momentaufnahme und der Betrachtung der Vergangenheit erfolgt die Durchführung einer Situationsanalyse, bei der z. B. die kamerabasierte Fußgängererkennung ermittelt, ob der Fußgänger den Fahrschlauch kreuzt und deshalb ein relevantes Objekt für den präventiven Fußgängerschutz wird. Besteht eine Gefahr, leitet die Assistenzfunktion weitere Schritte (wie eine Warnausgabe oder eine Bremsanforderung) an die dafür verantwortliche Komponente weiter. [Rei12]

Monokamera

Bei Systemen mit einer Monokamera wird nur eine Kamera in der Mitte der Windschutzscheibe montiert. Zur Objekterkennung gibt es unterschiedliche Ansätze, einer davon ist die bewegungsbasierte Methode. Hier wird aus dem optischen Fluss des Bildes die eigene Bewegung geschätzt und Bildregionen, bei denen ein abweichendes Bewegungsmuster auftritt, werden als Objekte identifiziert. Ein weiterer Ansatz ist die merkmalbasierte Methode, hier wird das Bild nach bestimmten Merkmalen wie Kanten, Ränder oder Symmetrien durchsucht und durch den Verlauf aus mehreren Bildern wird der Bewegungsverlauf berechnet. Der Nachteil liegt darin, dass Objekte nur dann erkannt werden, wenn die gesuchten Muster im Kamerabild vorhanden sind und dem System gelernt wurden. [Pra10]
Da nur ein zweidimensionales Bild von der Kamera erfasst wird, kann die longitudinale Entfernung zu Objekten nicht direkt bestimmt werden, sondern muss durch Modellannahmen (z. B. Fahrzeugbreite) geschätzt werden. Besonders die große Anzahl an unterschiedlichen Fahrzeugen, Fußgängern (Körpergröße, Bewegungsart) und anderen Objekten macht es schwierig, den genauen Abstand und die Relativgeschwindigkeit zu ermitteln. Der Vorteil liegt aber verglichen mit einem Stereokamera-System bei den geringeren Kosten durch nur eine Kamera und den geringeren Rechenaufwand zur Objektklassifikation. [Pra10], [Rei10]

Stereokamera

Bei Stereokamera-Systemen werden zwei Monokameras an der Windschutzscheibe mit einem definierten Abstand zueinander montiert. Aufgrund der leicht unterschiedlichen Blickwinkel der Kameras wird das menschliche Sehvermögen nachempfunden und dreidimensio-

nales Sehen ist damit möglich. Durch die gleichzeitige Aufnahme beider Kameras lässt sich direkt aus den beiden synchron aufgenommen Bildern die Tiefeninformation gewinnen, ohne dass über Modelle die Entfernung zu Objekten bestimmt werden muss. Somit kann der Abstand und die Relativgeschwindigkeit mit einer hohen Genauigkeit, verglichen mit einer Monokamera, bestimmt werden. [Fen10]
Die Schwierigkeit besteht aber bei der ständigen Kalibrierung der beiden Kameras zueinander, damit bei der Erstellung der Tiefeninformation keine Fehler einfließen. Zusätzlich sind laut [Pra10] Stereokameras bei der Bestimmung der Tiefeninformation empfindlicher in Bezug auf Lichtverhältnisse, Blendung und allgemeine Witterung und der Rechenaufwand ist verglichen mit einer Monokamera größer.

2.1.5.2 Anwendung Fußgängererkennung

Wie auch bei anderen optischen Sensoren schwankt die Leistung der Kamera mit der Sicht. Schlechte Wetterbedingungen oder eine schwache Belichtung führen zu einem starken Absinken der Reichweite und der Erkennungsrate. Ein großer Vorteil der Kamera ist die mögliche Objektklassifizierung, mit deren Hilfe Objekte wie Fußgänger, Fahrzeuge, Gegenstände oder auch Radfahrer klassifiziert werden. [Fen10] Aufgrund der hohen Pixeldichte des Kamerasensors ist verglichen mit anderen Sensoren eine hohe laterale und vertikale Winkelauflösung möglich. Besonders bei der Fußgängererkennung liegen somit die Vorteile bei der Kamera, da sie mit großer Sicherheit durch die Objekterkennung Fußgänger zuverlässig klassifiziert und eine ausreichende Reichweite von bis zu 50 m und einen Öffnungswinkel um die 40° erreicht. [Rei10]

2.2 Co-Simulation

Dieses Kapitel gibt einen Überblick über die Co-Simulation. Welche Probleme und Fehler bei der Co-Simulation entstehen und welche Methoden es gibt, um diese Probleme zu lösen, wird erläutert. Zum Abschluss sind Verweise auf Forschungsprojekte in der Automobilindustrie angeführt, bei denen die Co-Simulation zum Einsatz kommt.
In den letzten Jahrzehnten haben sich in der Automobilindustrie zahlreiche spezifische Simulationswerkzeuge für einzelne Fachgebiete etabliert. Disziplinübergreifende Betrachtungen von Gesamtsystemen sind dabei mit den Simulationswerkzeugen nur begrenzt möglich, da meist nur Systeme mit ähnlichem Zeitverhalten gemeinsam simuliert werden können (z. B. Thermomanagementsysteme mit einer heterogenen Tool-Landschaft). [BZWB12] Die Co-Simulation bietet die Möglichkeit, ein Gesamtsystem, bestehend aus mehreren Teilsystemen aus unterschiedlichen Disziplinen mit unterschiedlichen Zeitverhalten, gemeinsam zu simulieren (Multiraten-Multimethoden-Simulation) [BZWB12]. Als Beispiel hierfür ist die Gesamtsimulation des präventiven Fußgängerschutzes angeführt, hierbei kommt es zu

unterschiedlichen dynamischen Verhalten bei den einzelnen Teilsystemen (Mensch-Modell: niederfrequent, FEM-Modell Fahrzeug: hochfrequent).

Bei der Co-Simulation erfolgt die Simulation eines Gesamtsystems durch die Verwendung von unterschiedlichen Sprachen, Abstraktionsebenen, Tools oder Rechnersystemen in einer gemeinsamen Simulationsumgebung [KSW10], [GKL06]. Als Ergebnis können disziplinübergreifende Fragestellungen sowie Abhängigkeiten und Interaktionen zwischen einzelnen Teilsystemen besser betrachtet werden. Um dieses Ziel zu erreichen, hat die Co-Simulationsumgebung zwei Hauptaufgaben: Einerseits muss der Ablauf, zu welchem Zeitpunkt welches Teilsystem wann simuliert wird, koordiniert werden und andererseits muss der auftretende Datenaustausch zwischen den Teilsystemen gesteuert werden. Dazu müssen gegenseitige Abhängigkeiten zwischen den einzelnen Teilsystemen berücksichtigt werden und eine präzise Zusammenarbeit von mehreren Simulationswerkzeugen muss möglich sein, um nicht zusätzliche relevante Fehler in die Gesamtsimulation einzubringen. [BZWB12]

2.2.1 Aufbau einer Co-Simulation

Zum vereinfachten Verständnis der Simulation eines Gesamtsystems kann diese abstrakt aus mehreren Black-Boxes (Teilsysteme) betrachtet werden, die über eine Co-Simulationsplattform kommunizieren. Jede Black-Box verfügt dabei über ein Übertragungsverhalten und über Ein-/Ausgänge, um mit anderen Teilsystemen zu interagieren. Die Co-Simulations-plattform dient dabei als Verbindungsschnittstelle (Kopplung) zwischen den einzelnen Ein-/Ausgängen und steuert den Ablauf, wann welche Simulation ausgeführt wird. In Abbildung 2.14 ist der schematische Aufbau der Co-Simulationsumgebung ICOS (Independent Co-Simulation) [Vif13] dargestellt. Bei dieser Co-Simulationsumgebung steht noch ein grafisches Interface (Graphical User Interface) zur Verfügung, mit dem die Teilsysteme konfiguriert werden können (Simulationswerkzeug, Zeitverhalten usw.). Außerdem besteht die Möglichkeit der Konfiguration der Abhängigkeiten zwischen den einzelnen Teilsystemen. [BZWB12]
Die Co-Simulationsplattform muss definierte Schnittstellen („Wrapper“) für die Simulationswerkzeuge zur Verfügung stellen, damit der Datenaustausch und die Steuerung des Simulationswerkzeuges über die Co-Simulationsplattform möglich sind. Mit Hilfe von Wrapper können generische Schnittstellen realisiert werden und somit eine große Anzahl an Simulationswerkzeugen verwendet werden. Wrapper werden üblicherweise in C++ implementiert und nutzen Technologien wie COM oder TCP/IP für die Kommunikation. Damit ist es auch möglich, eine Client-Server-Architektur für die Co-Simulationsplattform zu verwenden, bei der die einzelnen Teilsysteme auf Clients ausgeführt werden und der Server für die Kopplung der Modelle sowie die Konfiguration und die Visualisierung der Ergebnisse zuständig ist.

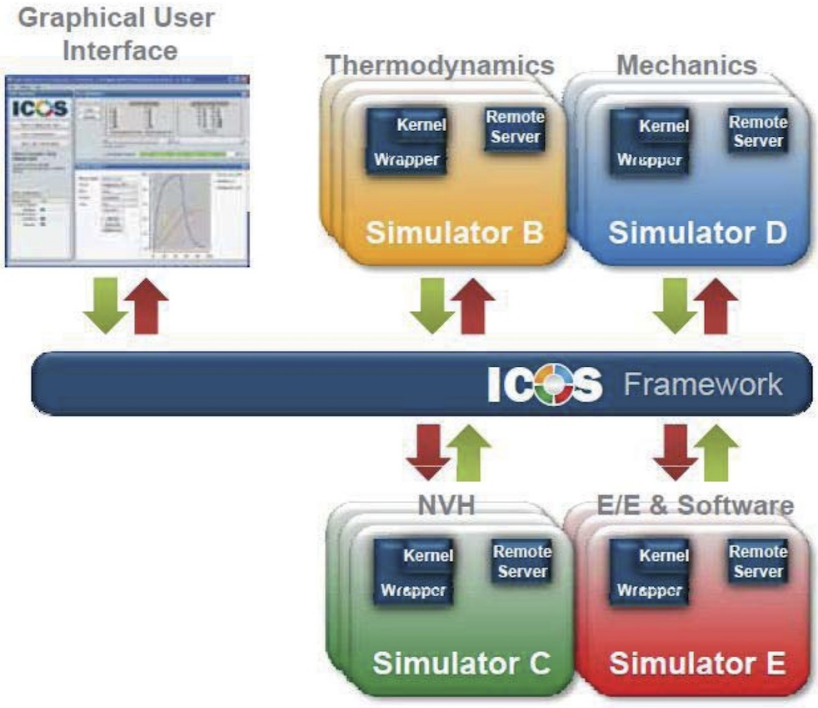

Abbildung 2.14: Schematische Darstellung der Co-Simulationsumgebung ICOS [BZWB12]

Um die Probleme gegenseitiger Abhängigkeit (interne Schleifen) und unterschiedlicher Zeitverhalten der Teilsysteme zu lösen, müssen intelligente Kopplungsmethoden und effiziente Ausführungsreihenfolgen von Simulationswerkzeugen verwendet werden. Auf diese Probleme und dafür mögliche Lösungswege wird in den folgenden Kapiteln näher eingegangen. [BZWB12]

2.2.1.1 Abhängigkeiten der Teilsysteme

Ein Problem bei komplexeren Gesamtsystemen sind die wechselseitigen Abhängigkeiten zwischen den miteinander verknüpften Teilsystemen. In den Makrozeitschritten ΔT kommt es zur Kopplung (Datenaustausch) zwischen den beiden Modellen. Die einzelnen Teilmodelle können dabei in unterschiedlichen Mikrozeitschritten δT simuliert werden. In Abbildung 2.15 ist ein Beispiel mit zwei Teilsystemen dargestellt, bei denen jeweils der Eingang eines Systems mit dem Ausgang des anderen Systems verbunden ist. Keines der beiden Modelle kann in der Gesamtsimulation alleine gelöst werden, dieses Problem wird als eine interne Schleife bezeichnet. [BZWB12] Das Problem kann durch zwei Methoden gelöst werden:

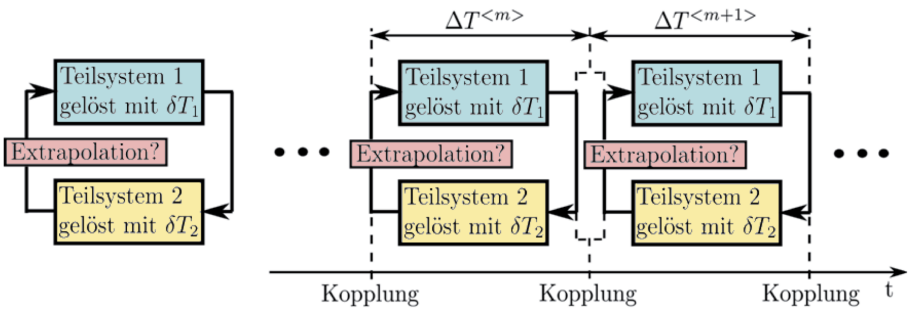

Abbildung 2.15: Interne Schleife bei einem System aus zwei Teilsystemen [BZWB12]

- **Iterative Kopplung**
  Jedes Teilsystem wird mehrfach für den aktuellen Makrozeitschritt (ΔT) gelöst und nach jeder Iteration kommt es zu einem Datenaustausch zwischen den Teilsystemen. Die unbekannten Koppelgrößen werden dabei beim ersten Iterationsschritt extrapoliert. Die Iterationen werden solange ausgeführt, bis ein definiertes Abbruchkriterium (z. B. Fehlerschranke) erreicht wird. [BZWB12]

- **Nichtiterative Kopplung**
  Bei dieser Methode wird jedes Teilsystem nur einmal pro Makrozeitschritt simuliert, jedoch wird dadurch ein Extrapolationsfehler in die Simulation eingebracht. Für die Extrapolation wird deshalb eine Zeithistorie der Koppelgröße verwendet. Bei dieser Methode können die Teilsysteme parallel oder sequentiell simuliert werden. Näheres dazu im Kapitel 2.2.1.2. [BZWB12]

Die Iterative Kopplung hat den Nachteil, dass mehrere Iterationen pro Makrozeitschritt durchgeführt werden müssen und somit das Simulationstool, die Möglichkeit bieten muss, Systemzustände des Modells und die Simulationszeit wieder zurückzusetzen. Diese Möglichkeit bieten nur wenige Simulationstools und somit ist die iterative Kopplung nur bedingt für die Co-Simulation einsetzbar. [BZWB12]

2.2.1.2 Ausführungsreihenfolge der Teilsysteme

Um eine effiziente Simulation des Gesamtsystems zu erzielen, ist eine Ausführung der Simulation der Teilsysteme in Signalflussrichtung zu bevorzugen. Kommt es aber zu internen Schleifen im Gesamtsystem, so ist bei der Wahl der Ausführungsreihenfolge zu beachten, dass es zu einer möglichst geringen Beeinflussung des Gesamtsystems kommt.
In Abbildung 2.16 ist ein Gesamtsystem mit zwei Teilsystemen (Regler und Strecke) dargestellt. Wenn beide Teilsysteme dasselbe Zeitverhalten aufweisen, stehen drei Möglichkeiten zur Ausführung der Teilsysteme zur Verfügung. In der linken Co-Simulation (a) erfolgt für die Eingangsgröße des dynamischen Systems eine Schätzung der Ausgangsgröße des

Reglers, mit diesem Wert erfolgt die Simulation des dynamischen Systems. Im nächsten Schritt wird mit der Ausgangsgröße des dynamischen Systems der Regler simuliert (ohne Extrapolation). Die beiden Teilsysteme werden nacheinander simuliert, deshalb wird dieser Ansatz als „sequentielle Ausführungsreihenfolge“ bezeichnet. Bei der Co-Simulation (b) erfolgt die Extrapolation und Simulation nach demselben Prinzip, jedoch in umgekehrter Reihenfolge. Auch hier muss nur eine Eingangsgröße geschätzt werden. Werden beide Teilsysteme parallel simuliert (siehe Abbildung (c)), so verringert sich die Simulationszeit. Da aber beide Eingangsgrößen extrapoliert werden müssen, führt dies zu einem größeren Kopplungsfehler. [BZWB12]

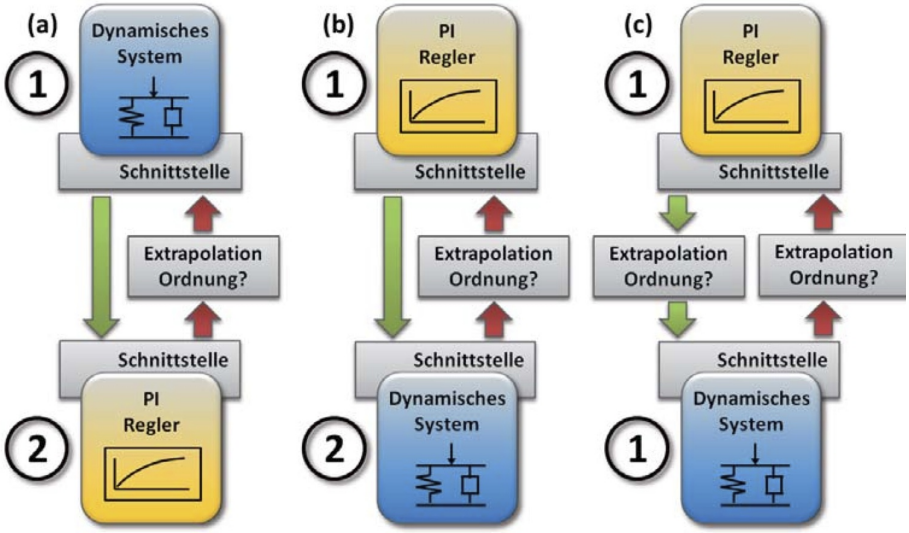

Abbildung 2.16: Möglichkeiten der Ausführungsreihenfolge bei der Co-Simulation mit zwei Teilsystemen [BZWB12]

Die parallele Ausführungsreihenfolge bietet den Vorteil der parallelen Simulation der Teilsysteme, jedoch muss die Makroschrittweite aufgrund des größeren Kopplungsfehlers verkleinert werden, um wiederum eine gewünschte Genauigkeit der Simulationsergebnisse zu erreichen. Auch sind Unstetigkeiten bei den Kopplungsgrößen meist problematisch, aus diesem Grund sind für die Co-Simulation sequentielle Ausführungsreihenfolgen besser geeignet. [BZWB12]

2.2.2 Extrapolationsverfahren

Wie im vorigem Kapitel erwähnt, müssen bei Vorhandensein von internen Schleifen Eingangsgrößen extrapoliert werden, damit die Simulation lösbar ist. Hierfür kommen am häufigsten die drei Extrapolationsmethoden: nullter Ordnung (zero-order-hold), erster Ord-

nung (first-order-hold) und zweiter Ordnung (second-order-hold) zum Einsatz. In Abbildung 2.17 sind die drei Methoden anhand eines Beispiels dargestellt.

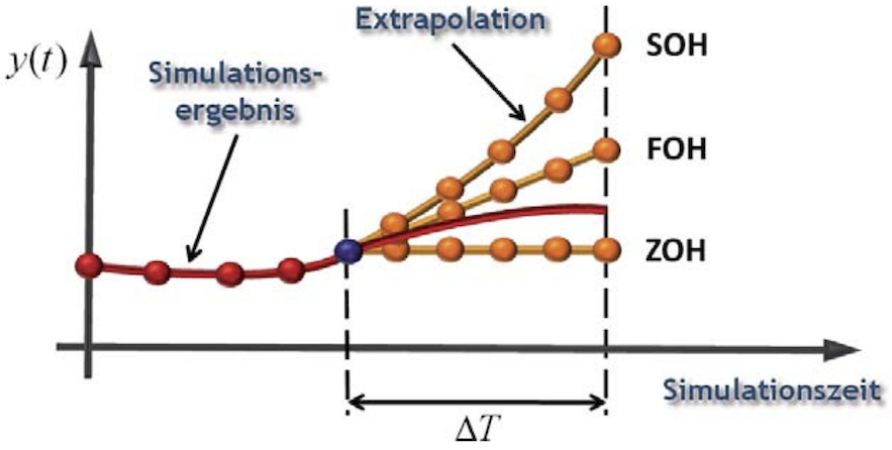

Abbildung 2.17: Möglichkeiten der Extrapolation [BZWB12]

Die einfachste Methode stellt die Extrapolation erster Ordnung dar, als Schätzwert wird einfach die Koppelgröße vom letzten Makrozeitschritt verwendet. Bei Systemen mit einer hohen Eigenfrequenz kommt es dadurch zu einer Limitierung der Ausgangsgröße und somit muss die Makroschrittweite verkleinert werden, um eine gewünschte Genauigkeit bei der Simulation zu erreichen. Abhilfe schaffen Extrapolationsmethoden mit einer Ordnung größer gleich eins, hierbei wird der Verlauf der Kopplungsgröße im nächsten Schritt mit einberechnet und dadurch können dynamischere Verläufe besser berücksichtigt werden. Extrapolationsmethoden mit einer Ordnung größer zwei kommen nicht zum Einsatz, da sich Unsicherheiten bei der Bestimmung der Ausgangsgröße verstärken, was zu stärkeren Schwankungen der Ausgangsgröße führt.

Ein Beispiel der Co-Simulation des oben angeführten Teilsystems mit unterschiedlichen Ausführungsreihenfolgen, Extrapolationsverfahren und Koppelschrittweiten ist in [BZWB12] näher erläutert.

2.3 Forschungsstand

Methode zur Berechnung der Feldeffektivität

In [Sra11] wird eine Methode zur Berechnung der Feldeffektivität integraler Sicherheitssysteme im Bereich des Fußgängerschutzes betrachtet. Dabei liegt das Hauptaugenmerk beim Funktionsentwicklungsprozess, damit eine Bewertung des Systems nach der Feldeffektivität, angefangen von der Systemidee bis zur Auslegung, durchgeführt werden kann.

Basierend auf der Unfalldatenbank wird die ursprüngliche Kollisionssituation automatisiert nachgebildet und mit unterschiedlichen Parametern simuliert, um die gesamte Unfallkette mit den Auswirkungen auf den Fußgänger zu betrachten. Um dieses Ziel zu erreichen, wurden zwei Methoden entwickelt: PreEffect-Open-Loop und PreEffect-Closed-Loop.
Bei der Methode PreEffect-Open-Loop erfolgt die Aufteilung der Simulation in zwei Prozessschritte. Im ersten Schritt wird das Unfallszenario mit unterschiedlichen Aktorauslösungen (Bremszeitpunkt, mit/ohne Bremsassistent) simuliert und die Ergebnisse werden in einer Datenbank archiviert. Im zweiten Schritt erfolgt die Simulation des Modelles mit der Sensorik und dem Funktionsalgorithmus, um die neue Unfallsituation zu erhalten. Danach wird auf die archivierten Daten mit der berechneten Aktuatorauslösung aus dem Unfallszenario (z. B. mit Bremsassistent) zurückgegriffen. Der Vorteil liegt darin, dass ein Unfallszenario mit unterschiedlichen Aktorauslösungen nur einmal simuliert werden muss und dadurch eine große Anzahl an Simulationen bei der Entwicklung des Funktionsalgorithmus oder Auslegung der Sensorik schnell durchgeführt werden kann. Der Nachteil liegt aber darin, dass keine Rückkopplung möglich ist. Soll z. B. das Bremssystem öffnen, wenn sich der Fußgänger nicht mehr im Gefahrenbereich befindet oder wenn die Kollision stattgefunden hat, kann mit dieser Methode nicht analysiert werden, da in der Datenbank mit der im Vorfeld simulierten Aktorauslösungen keine entsprechenden Daten vorliegen. Deshalb wurde ein zweiter Ansatz entwickelt, bei dem das komplette Unfallszenario in einem Schritt simuliert wird (PreEffect-Closed-Loop).

Simulation realer Unfälle mit aktiven Sicherheitssystemen
In der Arbeit von [Erb09] wird eine Methode beschrieben, die anhand realer Unfälle die Auswirkung aktiver Sicherheitssysteme (Kollisionsvermeidung mit anderen Fahrzeugen) betrachtet. Basierend auf der GIDAS-Datenbank (German In-Depth Accident Study), in der Unfälle dokumentiert und rekonstruiert sind, erfolgt die Betrachtung eines Szenarios ohne aktive Sicherheitssysteme und mit aktiven Sicherheitssystemen anhand unterschiedlichen Parametern.
Zur Simulation des Unfallherganges wird der CarMaker von IPG verwendet, dazu wird dieser mit den ermittelten Daten aus der Unfallstatistik initialisiert. Im ersten Schritt erfolgt die Simulation des realen Szenarios ohne Sicherheitssystem, danach wird die Simulation mit dem realen Unfall verglichen. Im nächsten Schritt werden Sicherheitssysteme hinzugefügt und in mehreren Durchläufen mit unterschiedlichen Parametern simuliert. Nach den Simulationen werden die Ergebnisse der einzelnen Simulationen verglichen und die Auswirkungen (Aufprallgeschwindigkeit, Positionen) von den aktive Sicherheitssysteme können ermittelt werden.
Zur Berechnung der Sichtbarkeit der Sensoren wird ein geometrisches Modell verwendet, mit dem es möglich ist, Sensoren mit einer Reichweite, einem Öffnungswinkel und einer

Latenz nachzubilden. Der Bewegungsverlauf der beteiligten Fahrzeuge ist im CarMaker realisiert. Der Simulationsvorgang endet bei der Kollision und wird danach nicht mehr weitergeführt. Eine weitere Analyse der Unfallschwere für die Insassen ist nicht möglich, da der Schwerpunkt in dieser Arbeit nur bis zur Betrachtung des Kollisionszeitpunktes liegt.

Co-Simulation in der Fahrzeugentwicklung
In der Fahrzeugentwicklung ist der Ansatz bereits bekannt und findet in den unterschiedlichsten Bereichen Verwendung. Als Beispiel wird in [KSW10] eine Co-Simulationsplattform für die Modellierung mechatronischer Systeme dargestellt. Hier wird ein Anwendungsfall anhand eines Drive-By-Wire Systems gezeigt. In [Zeh11] wird die Co-Simulation für die Bewertung eines Energiemanagementsystems für ein Hybrid-Elektrofahrzeug eingesetzt. Eine mit Hilfe der Co-Simulation durchgeführte Analyse der Stabilität von einem Fahrzeugboardnetz mit zwei Spannungen ist in [WZ12] beschrieben. Ein weiteres Beispiel für die Verwendung der Co-Simulation im Automobilbereich ist das Functional Mockup Interface [Blo12], welches im Rahmen des MODELISAR Projektes entwickelt wurde. Das Interface wurde gemeinsam mit Partnern aus verschiedenen Teilen der Automobilindustrie (OEMs, Tier x, Tool-Herstellern) entwickelt.

2.4 Unfallszenarien mit Fußgängern

Die meisten Fußgängerunfälle in Verbindung mit einem Fahrzeug passieren, wenn ein Fußgänger versucht, vor einem herankommenden Fahrzeug die Straße zu überqueren. Dies geht aus der Unfallstatistikdatenbank GIDAS hervor. [Dom12]
In Abbildung 2.18 wird die Verteilung der Unfallszenarien dargestellt. Zählt man die beiden häufigsten Szenarien zusammen, bei dem ein Fußgänger die Straße quert, so kommt man bereits auf 74,3 % aller Fußgängerunfälle in Verbindung mit einem PKW. Deshalb wird für die Betrachtung des Gesamtsystems das Szenario mit einem überquerenden Fußgänger gewählt, der erst kurz vor dem Zusammenstoß mit dem Fahrzeug für den Fahrer sichtbar ist.

Beim betrachteten Szenario bewegt sich das Ego-Fahrzeug mit konstanter Geschwindigkeit auf den Fußgänger zu, welcher die Straße mit konstanter Geschwindigkeit von rechts zum Fahrzeug überquert. Die Dauer von der ersten Sichtbarkeit bis zur Kollision kann einerseits durch den Aufprallpunkt am Ego-Fahrzeug festgelegt werden. Andererseits ist es auch möglich, die Sichtbarkeit durch den Abstand vom Verdeckungsfahrzeug zum Fahrbahnrand zu variieren. Der Fußgänger ist umso länger sichtbar, je weiter links er am Fahrzeug auftrifft, da er einen längeren Weg von der Verdeckung bis zum Aufprallpunkt zurücklegen muss.

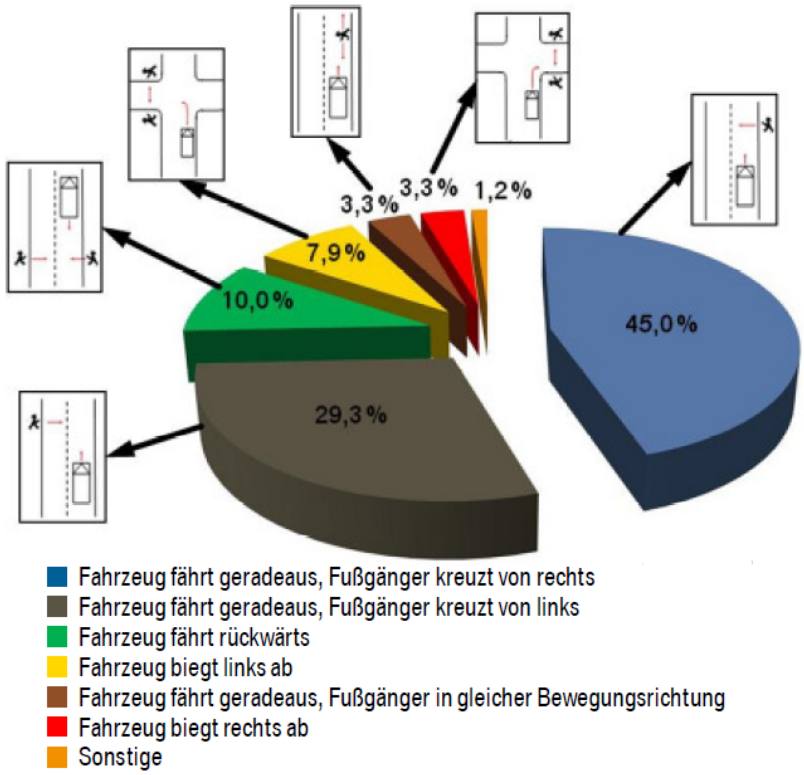

Abbildung 2.18: Unfallstatistik Fußgängerunfälle, [Dom12]

Betrachtet man das dargestellte Szenario in Abbildung 2.19, so werden für die Simulation folgende Modelle benötigt:

1. Ego-Fahrzeug
2. Fußgänger
3. Verdeckungsfahrzeug am Straßenrand
4. Sichtbarkeitsmodell

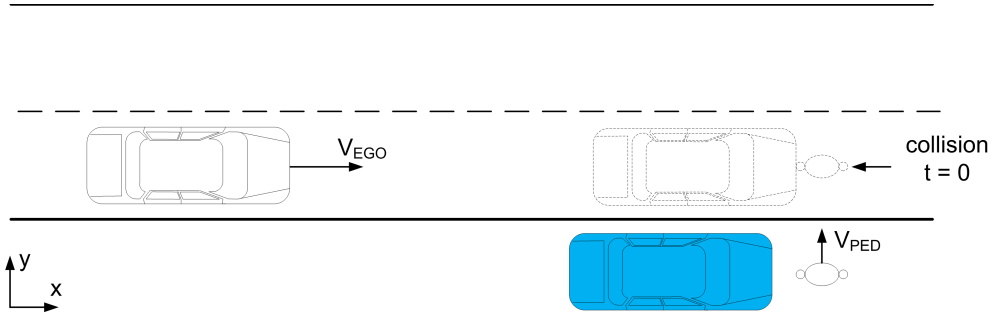

Abbildung 2.19: Darstellung des ausgewählten Szenarios [SKW13]

Auf Details zu den einzelnen Teilmodellen und wie sie miteinander verbunden sind wird im Kapitel 3 Simulation näher eingegangen.

Mögliche Einflussgrößen beim Szenario:

- Geschwindigkeit Ego-Fahrzeug
- Geschwindigkeit Fußgänger
- Abstand Verdeckungsfahrzeug - Fahrbahnrand
- Abstand Verdeckungsfahrzeug - Fußgänger
- Aufprallpunkt Fußgänger - Ego-Fahrzeug

Die beiden Systeme (Kamera und Ko-TAG) werden bei unterschiedlichen Geschwindigkeiten beider Teilnehmer und Abständen des Verdeckungsfahrzeuges zur Fahrbahn vermessen und analysiert. Genauere Details zu den einzelnen Messungen sind im Kapitel 5 angeführt.

Kapitel 3

Simulation

Für die Simulation des präventiven Fußgängerschutzes wird die Co-Simulationsumgebung ICOS [Vif13] genutzt. ICOS bietet die Möglichkeit, unterschiedliche Teilmodelle gemeinsam zu simulieren, wobei die Modelle nicht in der gleichen Entwicklungsumgebung realisiert sein müssen. In dieser Arbeit wurden aber alle Teilmodelle in Matlab Simulink [Mat13] umgesetzt.
Die Co-Simulation besteht aus dem Ego-Fahrzeug, Fußgänger, Verdeckungsfahrzeug und Sichtbarkeitsmodell. Das Ego-Fahrzeug wird dabei unterteilt in das Längsdynamik-Modell, den Sensor zur Fußgängererkennung und den Funktionsalgorithmus für die Warn- und Bremslogik.
Für das Modell des Fußgängers, des Verdeckungsfahrzeuges und das Längsdynamik-Modell des Ego-Fahrzeuges wurden vorhandene Modelle verwendet, beim Längsdynamik-Modell wurde nur der Bremsvorgang neu modelliert. Die anderen Modelle wurden in dieser Arbeit entwickelt. In Abbildung 3.1 sind die Abhängigkeiten zwischen den einzelnen Modellen dargestellt und folgende Liste gibt eine Übersicht aller beteiligten Modelle:

- Ego-Fahrzeug:
  - Längsdynamik-Modell
  - Sensor
  - Funktionsalgorithmus
- Fußgänger
- Verdeckungsfahrzeug
- Sichtbarkeitsmodell

Durch die Unterteilung in mehrere Modelle, die miteinander Daten austauschen, ist es möglich, diese durch andere Modelle zu ersetzen. Für die kamera- und Ko-TAG-basierte Fußgängererkennung kann die gleiche Simulation verwendet werden, nur das Sensor-Modell ist zu ersetzen.

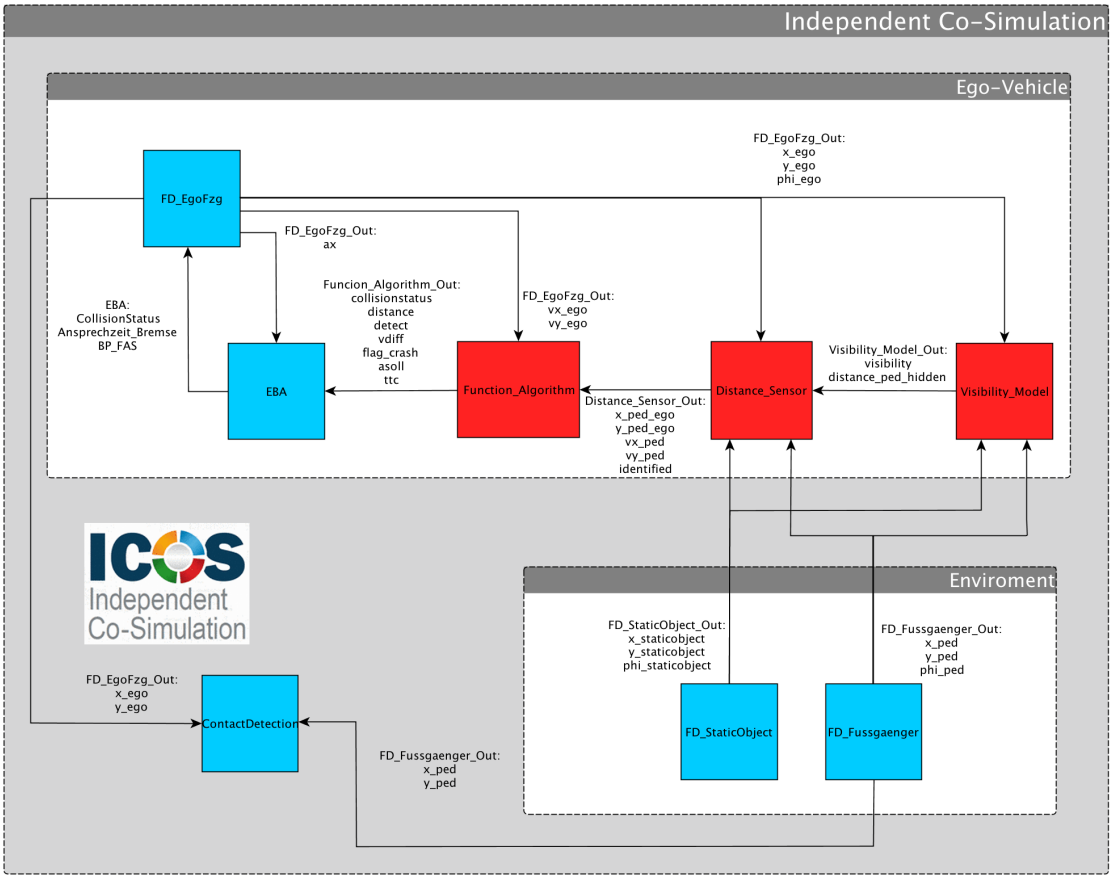

Abbildung 3.1: Übersicht der Gesamtsimulation mit ICOS

Aus der Konfiguration des Szenarios und den Kenngrößen des Fahrzeuges, wie z. B. den Sensoreigenschaften oder der Warn- und Bremskonfiguration, ergibt sich für die Simulation folgende Liste an Parametern, die definierbar sind:

Ego-Fahrzeug:

- Startposition
- Länge/Breite
- Fahrzeugnullpunkt
- Drehrichtung
- Geschwindigkeit in x- und y-Richtung
- Totzeit bis zum Start der Bremsung
- Verzögerungsaufbau

Fußgänger:

- Startposition
- Geschwindigkeit in x- und y-Richtung

Verdeckungsfahrzeug:

- Position
- Länge/Breite
- Drehrichtung

Sensor:

- Öffnungswinkel
- Reichweite
- Framerate
- Verbauort in x- und y-Richtung
- Dauer bis zur Identifizierung oder Anzahl der Frames pro Sekunde

Funktionsalgorithmus:

- Fahrschlauchbreite
- Warn- und Bremskorridor (TTC, Abstand zum Fahrschlauch)
- Sollverzögerung

3.1 Modellierung Fahrzeugumgebung

3.1.1 Fußgänger

Um den Fußgänger zu modellieren, wird ein vorhandenes Fußgängermodell verwendet. Das Modell besteht aus einem Massepunkt, der sich mit konstanter Geschwindigkeit linear im Koordinatensystem bewegen kann. Beim betrachteten Lastfall bewegt sich der Fußgänger mit einer konstanten Geschwindigkeit orthogonal zum Ego-Fahrzeug und reagiert nicht auf externe Einflüsse, wie das herannahende Ego-Fahrzeug. Dafür kann das vereinfachte in Matlab Simulink erstellte Modell des Fußgängers verwendet werden.
Durch die Startposition *[target_x0, target_y0, target_z0]*, den Winkel *target_alpha* und die Geschwindigkeit *target_v0* wird der Bewegungsverlauf des Fußgängers festgelegt.

Initialisierungsparameter:

```
target_x0 = 0.0 %[m] x-Position
target_y0 =  3.178 %[m] y-Position
target_z0 = 0.5 %[m] z-Position (irrelevant)
target_alpha = -1.57079632679 %[rad] Bewegungsrichtung
target_v0 = 1.4 %[m/s] Geschwindigkeit
```

3.1.2 Verdeckungsfahrzeug

Mit Hilfe des Verdeckungsfahrzeuges am Straßenrand kann der Fußgänger für das Ego-Fahrzeug verdeckt werden und ist somit erst optisch sichtbar, wenn dieser aus dem Schatten des Verdeckungsfahrzeuges hervorkommt. Ohne das Verdeckungsfahrzeug wäre der Fußgänger wesentlich früher sichtbar und besonders bei der Betrachtung der Kamera, würde die Zeit, bis der Fußgänger als solcher erkannt wird, vernachlässigbar sein.
Für das Verdeckungsfahrzeug wird das gleiche Modell wie für das fahrende Fahrzeug verwendet, mit dem Unterschied, dass die Geschwindigkeit des Verdeckungsfahrzeuges gleich null ist.
Über den Abstand zum Fahrschlauch und den Abstand zwischen Verdeckungsfahrzeug und Fußgänger kann der Zeitpunkt variiert werden, ab wann der Fußgänger für das Ego-Fahrzeug sichtbar ist.

Initialisierungsparameter:

```
AccPedal_init = 0.00 %[-] Pedalstellung
x0_ini = -3 %[m] x-Position Fahrzeugnullpunkt
y0_ini = -5 %[m] y-Position Fahrzeugnullpunkt
z0_ini = 0.2337 %[m] z-Position Fahrzeugnullpunkt (irrelevant)
phixB_ini = 0 %[rad] Drehrichtung in x
phiyB_ini = 0 %[rad] Drehrichtung in y
phizB_ini = 0 %[rad] Drehrichtung in z (irrelevant)
vxB_ini = 0 %[m/s] Geschwindigkeit in x-Richtung
vyB_ini = 0 %[m/s] Geschwindigkeit in y-Richtung
vzB_ini = 0 %[m/s] Geschwindigkeit in z-Richtung (irrelevant)
```

3.2 Sichtbarkeitsmodell

Damit die Kamera einen Fußgänger überhaupt erkennen kann, muss eine direkte Sichtverbindung vorhanden sein. Hindernisse, die sich zwischen Kamera und Fußgänger befinden, müssen erkannt und daraus die Sichtbarkeit des Sensors berechnet werden. Dafür wird das Sichtbarkeitsmodell verwendet, es berechnet, ob sich der Fußgänger im Blickfeld der Kamera befindet, unabhängig von den Kameraparametern (Reichweite und Öffnungswinkel), wie in Abbildung 3.2 dargestellt ist. Die Entfernung und der Winkel des Fußgängers zum Fahrzeug werden durch das Modell nicht berechnet, es liefert nur die Sichtbarkeit. [?]
Zur Vereinfachung gilt ein Fußgänger als sichtbar, wenn sich der Mittelpunk des Fußgängers 20 cm (halbe Fußgängerbreite) im Blickfeld der Kamera befindet. Die Kontur des Fahrzeuges wird nicht berücksichtigt, da eine Vielzahl der verwendete Algorithmen erst funktionieren, wenn der Fußgänger voll sichtbar ist.

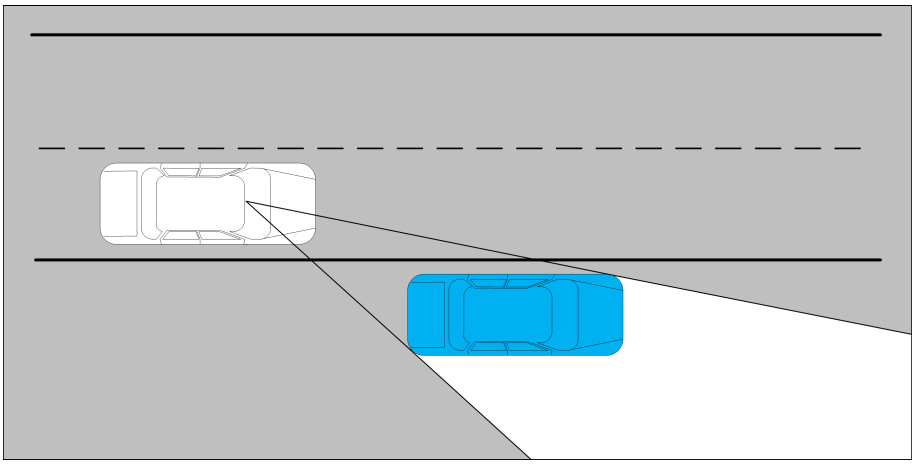

Abbildung 3.2: Visualisierung der Sichtbarkeit [SKW13]

Adaption für Ko-TAG:
Der Ko-TAG-Sensor bietet gegenüber der Kamera den Vorteil, dass man mit dem Sensor den Fußgänger bereits erkennen kann, wenn dieser noch nicht sichtbar ist. Aus diesem Grund liefert das Sichtbarkeitsmodell noch den zusätzlichen Parameter, wie weit sich der Fußgänger hinter der Fahrzeugkante des Verdeckungsfahrzeuges befindet.

Realisierung:
Für die Realisierung des Sichtbarkeitsmodells wurde Matlab Simulink gewählt. Aufgrund der flexiblen Möglichkeit der Realisierung, der Verknüpfung mit anderen Modellen und der Möglichkeit der Einbindung des Modells in ICOS, wurde Simulink verwendet.
Das Sichtbarkeitsmodell (Abbildung 3.3) berechnet zuerst aus den absoluten Positionen, die benötigten Relativabstände zwischen dem Sensor, dem Verdeckungsfahrzeug (Static Object) und dem Fußgänger (Ped).
Im Fuktionsblock *Calculate_ Nearest_ Corner_ Static_ Object* wird der maximal sichtbare Öffnungswinkel aus der Sicht des Sensors bestimmt. Aus der Sensorposition, dem Nullpunkt des Verdeckungsfahrzeuges und der Drehrichtung des Fahrzeuges wird die Fahrzeugkante berechnet, welche die Sicht am meisten einschränkt und daraus wird der sichtbare Öffnungswinkel berechnet.
Zuerst erfolgt die Berechnung der vier Eckpunkte des Verdeckungsfahrzeuges aus dem Fahrzeugnullpunkt, im nächsten Schritt erfolgt mit Hilfe einer Rotationsmatrix die Drehung der Eckpunkte mit der Drehrichtung des Verdeckungsfahrzeuges. Die Eckpunkte werden um den Abstand zwischen Nullpunkt des Verdeckungsfahrzeuges und der Sensorposition verschoben, danach werden die Winkel zu den vier Fahrzeugkanten berechnet. In Abhängigkeit der Position des Verdeckungsfahrzeuges (steht es links oder rechts vom Ego-Fahrzeug) wird

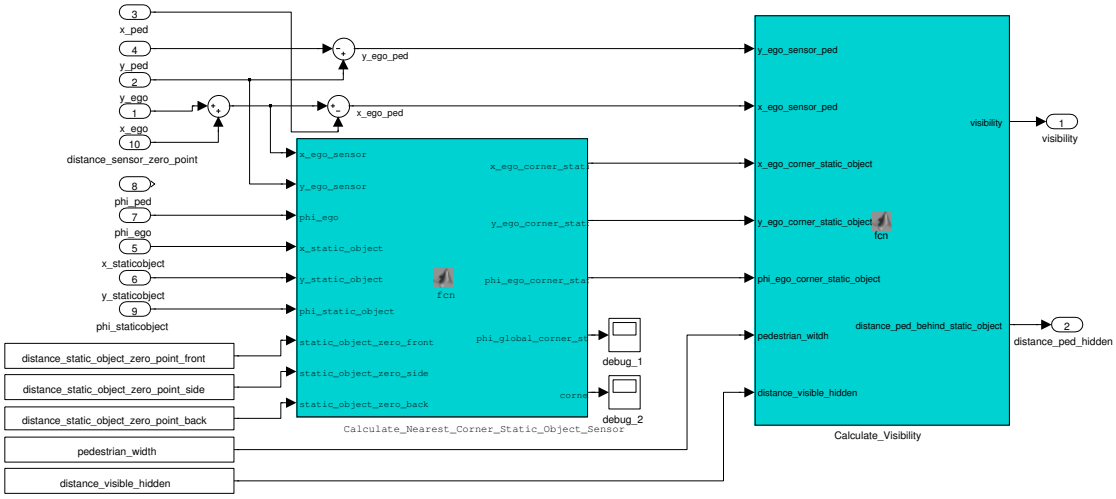

Abbildung 3.3: Implementierung des Sichtbarkeitsmodell in Matlab Simulink

die Fahrzeugkante bestimmt, die die Sicht des Sensors am meisten einschränkt. Aus der Position der Fahrzeugkante und der des Sensors kann der Öffnungswinkel bestimmt werden. Der Funktionsblock *Calculate_ Visibility* berechnet daraus mit der Position des Fußgängers, ob dieser für den Sensor sichtbar ist oder nicht. Der Fußgänger gilt als sichtbar, wenn sich sein Mittelpunkt mit einem Abstand von *pedestrian_ witdh/2* zum Öffnungswinkel der davor verdeckten Fahrzeugkante befindet.
Über den Parameter *distance_ visible_ hidden* lässt sich definieren, ab wann das Modell den Fußgänger als verdeckt erkennt. Der Parameter definiert den Abstand zwischen dem Mittelpunkt des Fußgängers und der betroffenen Fahrzeugkante. Befindet sich der Fußgänger innerhalb dieses Bereiches, so liefert das Modell bei der Ausgangsgröße *Visibility* den Wert 0,5 und bei der Ausgangsgröße *distance_ ped_ behind_ static_ object* den Abstand des Fußgängers zur Fahrzeugkante des Verdeckungsfahrzeuges.

Initialisierungsparameter:

```
%Init values for Visibility_Model
distance_sensor_zero_point = 2.33; %[m]  Position des Sensors zum Ego-Fahrzeug-Nullpunkt
distance_static_object_zero_point_side = 0.93;  %[m] Distanz Verdeckungsfahrzeug vordere Seite
distance_static_object_zero_point_back = 2.57;  %[m] Distanz Verdeckungsfahrzeug hintere Stoßstange
distance_static_object_zero_point_front = 2.33; %[m] Distanz Verdeckungsfahrzeug vordere Stoßstange
pedestrian_width = 0.4; %[m]
distance_visible_hidden = 1; %[m]
```

Eingangsgrößen:

```
x_ego %[m] x-Position Ego-Fahrzeug
y_ego %[m] y-Position Ego-Fahrzeug
x_ped %[m] x-Position Fußgänger
```

```
y_ped %[m] y-Position Fußgänger
phi_ego %[rad] Ausrichtungswinkel des Ego-Fahrzeug
x_staticobject %[m] x-Position Verdeckungsfahrzeug
y_staticobject %[m] y-Position Verdeckungsfahrzeug
phi_staticobject %[rad] Ausrichtungswinkel Verdeckungsfahrzeug
```

Ausgangsgrößen:

```
visibility %[-] Sichtbarkeit: 0,0.5,1
distance_ped_hidden %[m] Abstand zwischen Fußgänger und Verdeckungsfahrzeug
```

3.3 Modellierung Kamera

Das Kameramodell ist für die Identifizierung des Fußgängers verantwortlich. Das Modell wertet die prinzipielle Sichtbarkeit vom Sichtbarkeitsmodell (Kapitel 3.2) aus und überprüft ob der Fußgänger im Erfassungsbereich der Kamera liegt. Wenn ein Fußgänger als solcher identifiziert wird, liefert das Modell die relative Position zwischen Ego-Fahrzeug und Fußgänger sowie die absolute Geschwindigkeit des Fußgängers in x- und y-Richtung. In Abbildung 3.4 ist die Sichtbarkeit der Kamera mit definiertem Öffnungswinkel, Reichweite und der prinzipiellen Sichtbarkeit dargestellt.
Aufgrund der abgeleiteten Anforderungen an das Kameramodell können folgende Größen beim Kameramodell verändert bzw. vorgegeben werden: Framerate, Identifizierungszeit, aufschlagender Fehler, Öffnungswinkel, Reichweite und Verbauort.

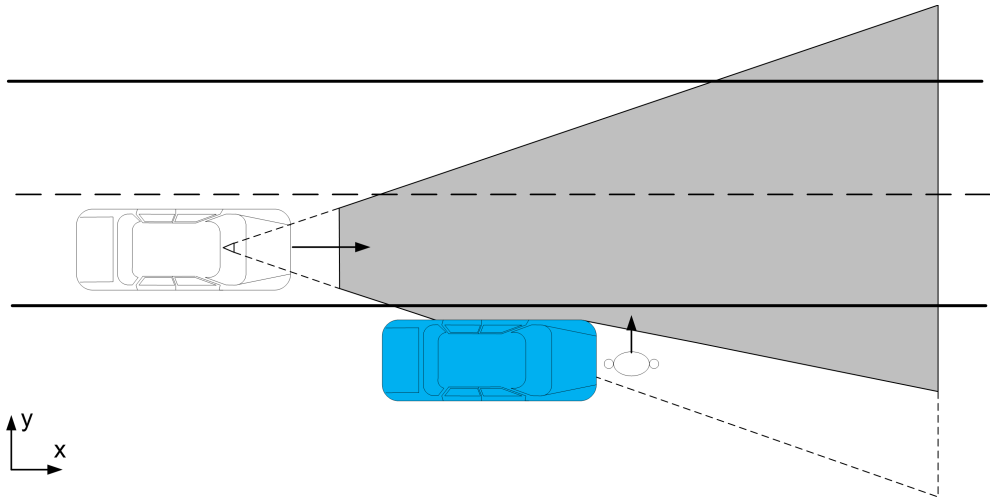

Abbildung 3.4: Sichtbereich des Kameramodells [SKW13]

Realisierung:
Das Kameramodell wurde in Matlab Simulink realisiert und ist in Abbildung 3.5 dargestellt. Das Modell besteht im Wesentlichen aus drei Teilblöcken für die Berechnung der Sichtbarkeit, die Geschwindigkeitsschätzung und die Identifikation des Fußgängers.
Im ersten Schritt wird überprüft, ob der Fußgänger für die Kamera überhaupt sichtbar ist. Dafür wird die *visibility* (Ausgangsgröße vom Sichtbarkeitsmodell) ausgewertet und im Funktionsblock *FB_ Visibility_ Sensor* wird berechnet, ob der Fußgänger innerhalb der Kamerareichweite (*camera_ sensor_ distance*) und des Öffnungswinkels (*camera_ sensor_ angle*) liegt. Bei der Berechnung wird die Position der Kamera (*distance_ camera_ zero_ point*) berücksichtigt, da es einen Unterschied gibt, ob der Sensor sich vorne an der Stoßstange oder bei der Windschutzscheibe befindet. Zusätzlich berechnet der Funktionsblock, ob sich der Fußgänger links oder rechts vom Fahrzeug befindet. Für einen Fehleraufschlag bei der Entfernungsmessung in y-Richtung muss berücksichtigt werden, auf welcher Seite sich der Fußgänger befindet. Sowohl für die Entfernungsmessung in x-Richtung als auch in y-Richtung kann ein Fehler in den Look-Up-Tabellen *camera_ error_ x.mat und camera_ error_ y.mat* angegeben werden.

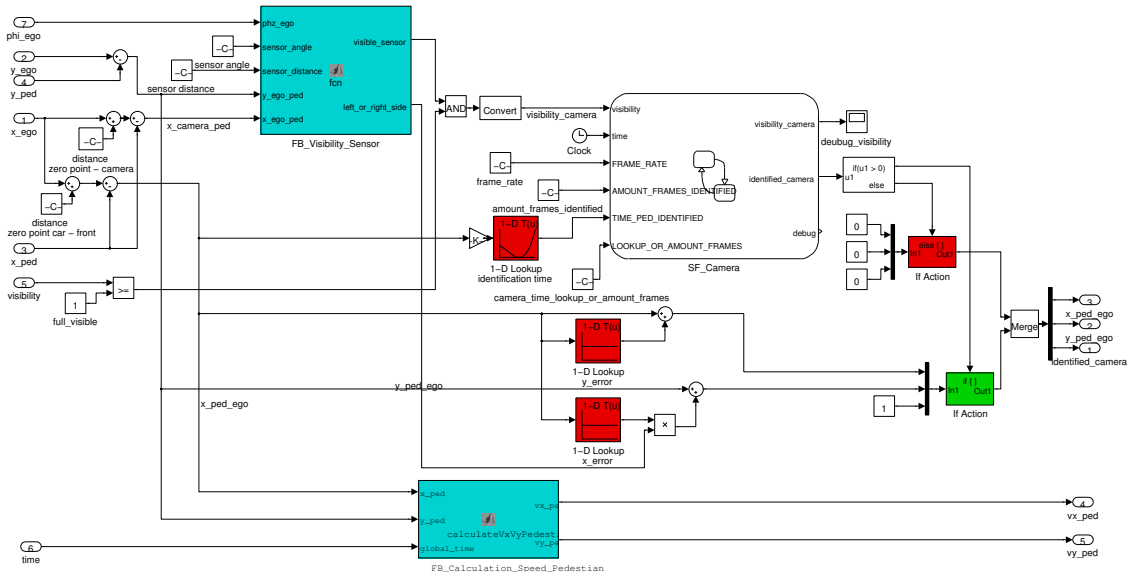

Abbildung 3.5: Implementation des Kameramodells in Matlab Simulink

Wenn der Fußgänger für die Kamera sichtbar ist, gilt er noch nicht als identifiziert, da die Kamera nicht sofort die Parameter des Fußgängers (Position und Geschwindigkeit) bestimmen kann, sie benötigt abhängig von der Situation unterschiedlich lange zur Bestimmung der Parameter. Die Zeit bis zur Identifizierung des Fußgängers kann fest oder abhängig von der Entfernung eingestellt werden. Durch die Parameter *camera_ time_ lookup_ or_ amount_ frames = 0* und z. B. *camera_ amount_ frames_ identified = 3*, kann die Dauer bis zur

Identifizierung auf 3 Frames eingestellt werden. Dadurch ist es auch möglich einen idealen Sensor nachzubilden. Wird der Parameter *camera_time_lookup_or_amount_frames* auf eins gesetzt, so greift das Modell auf die Look-Up-Tabelle *time_to_identified.mat* zurück, bei der in Abhängigkeit der Distanz zwischen Kamera und Fußgänger die Identifikationsdauer gespeichert ist.
Die Zustandsmaschine *SF_Camera* (Abbildung 3.6) ist für die Identifikation des Fußgängers zuständig. Wenn für die Kamera kein Fußgänger sichtbar ist, bleibt die Zustandsmaschine im Zustand **WAIT**. Sobald ein Fußgänger sichtbar ist, wird in den Zustand **VISIBLE_FIRST_TIME** gewechselt und der Zeitpunkt der ersten Sichtbarkeit gespeichert. Solange der Fußgänger aufgrund der benötigten Identifikationszeit noch nicht identifiziert ist, bleibt die Zustandsmaschine im Zustand **WAIT_UNTIL_IDENTIFIED**, danach wird in den Zustand **IDENTIFIED** gewechselt.

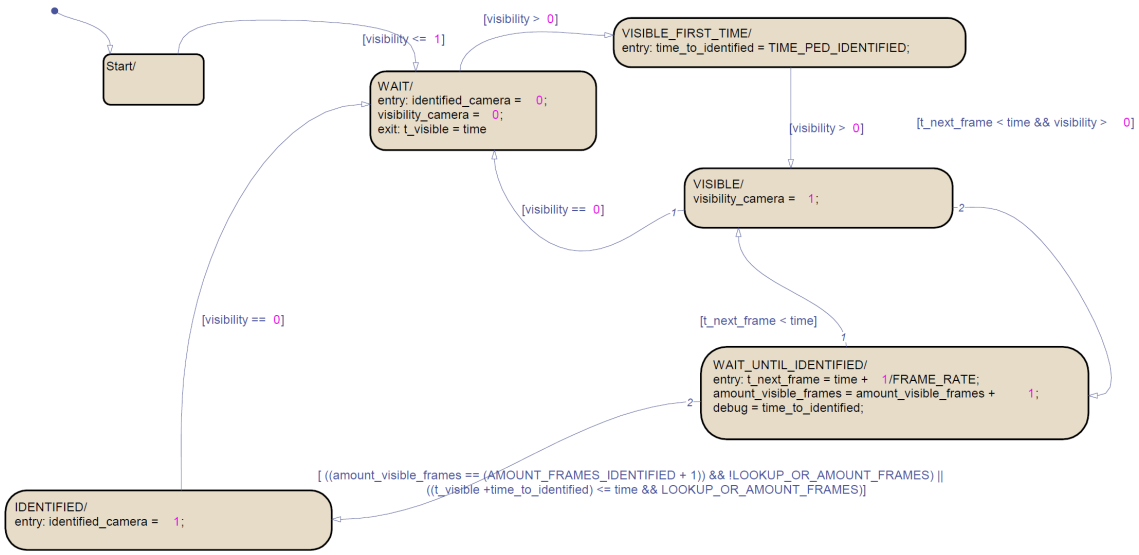

Abbildung 3.6: Zustandsmaschine des Kameramodell zur Idnetifizierung des Fußgängers

Das Kameramodell berechnet die Geschwindigkeit des Fußgängers in x- und y-Richtung im Funktionsblock *FB_Calculation_Speed_Pedestrian*. Dafür wird alle 100 ms die aktuelle Geschwindigkeit bestimmt, diese errechnet sich aus der aktuellen Position und der vor 100 ms. Das Modell bestimmt aus den letzten drei gemessenen Geschwindigkeiten die aktuell geschätzte Geschwindigkeit des Fußgängers, um beim Geschwindigkeitsverlauf starke Beschleunigungen zu glätten. Der Einfluss der zuletzt gemessenen Geschwindigkeit ist am größten und fällt mit jeder weiteren Messung ab. [Chr09]
Wenn die Zustandsmaschine den Fußänger identifiziert hat, dann setzt das Kameramodell die Ausgangsgröße *identified_camera* auf eins und die Parameter des Fußgängers (*x_ped_ego, y_ped_ego, vx_ped, vy_ped*) werden ausgegeben.

Initialisierungsparameter:

```
%Init values for Camera_Model
distance_zero_point_front = 2.33;  %[m] Distanz Nullpunkt - Front
distance_camera_zero_point = 0.31; %[m]  Position des Sensors zum Ego-Fahrzeug-Nullpunkt
camera_sensor_angle = 19; %[∞] Öffnungswinkel Kamera 1.Seite
camera_sensor_distance = 40; %[m] Reichweite
camera_frame_rate = 12; %[Hz] Abtastrate
camera_amount_frames_identified = 4; %[n] Anzahl der Frames bis der Fußgänger identifiziert ist
camera_time_lookup_or_amount_frames = 0; %[-]   1: Identification time = look-up-table
                              0: Identification time = amount_frames * camera_frame_rate,
load('time_to_identified.mat'); %Lade Identifikationszeit
camera.v1 = time_to_identified(:,[1]);
camera.v2 = time_to_identified(:,[2]);

load('camera_error_x.mat'); %Lade x-y-Fehler
load('camera_error_y.mat');
```

Eingangsgrößen:

```
x_ego %[m] x-Position Ego-Fahrzeug
y_ego %[m] y-Position Ego-Fahrzeug
x_ped %[m] x-Position Fußgänger
y_ped %[m] y-Position Fußgänger
visibility %[-] Sichtbarkeit des Fußgängers
time %[s] Globale Zeit
phi_ego %[rad] Ausrichtungswinkel Ego-Fahrzeug
```

Ausgangsgrößen:

```
identified_camera %[-] Sichtbarkeit: 0,1
x_ped_ego %[m] x-Abstand zwischen Fußgänger und Ego-Fahrzeug
y_ped_ego %[m] y-Abstand zwischen Fußgänger und Ego-Fahrzeug
vx_ped %[m/s] Geschwindigkeit des Fußgängers in x-Richtung
vy_ped %[m/s] Geschwindigkeit des Fußgängers in y-Richtung
```

3.4 Modellierung Ko-TAG

Da der Fußgängerschutz nicht nur mit der Kamera, sondern auch mit der Ko-TAG-basierten Fußgängererkennung durchgeführt werden soll, wird auch für die Modellierung ein leicht abgewandeltes Modell der kamerabasierten Fußgängererkennung verwendet. Gleich wie das Kameramodell ist auch das Ko-TAG-Modell für die Positionsbestimmung des Fußgängers zuständig. Im Gegensatz zur Kamera kann der Fußgänger bereits mit dem Ko-TAG-Sensor identifiziert werden, wenn dieser optisch noch nicht sichtbar ist. Dieser Umstand wird im Modell dahingehend berücksichtigt, dass der Abstand, wie weit der Fußgänger verdeckt ist, miteinbezogen wird. Das Modell stellt die gleiche Schnittstelle wie das Kameramodell zur Verfügung und liefert bei einer Identifizierung des Fußgängers dessen Position und Geschwindigkeit. Damit ist es in der Simulation einfach möglich, die beiden Sensoren auszutauschen und in der gleichen Simulationsumgebung zu vergleichen. Ähnlich wie bei der

Kamera können folgende Parameter vorgegeben werden: Framerate, Öffnungswinkel, Reichweite, Verbauort, aufschlagender Fehler bei Verdeckung und freier Sicht auf den Fußgänger.

Realisierung:
In Matlab Simulink wurde das Modell des Ko-TAG-Sensors realisiert und ist in Abbildung 3.7 dargestellt. Das Modell verwendet die Funktionsblöcke *FB_ Visibility_ Sensor* und *FB_ Calculation_ Speed_ Pedestrian* vom Kameramodell. Der Hauptunterschied liegt in der Identifizierung des Fußgängers. Bei der Kamera ist der Fußgänger bei vollständiger Sichtbarkeit erst nach einer gewissen Zeit identifiziert, dafür wird die Zustandsmaschine *SF_ Camera* verwendet. Erste Messungen mit dem Ko-TAG-Sensor haben gezeigt, dass der Fußgänger wesentlich früher erkannt wird, auch wenn er voll verdeckt ist. Deshalb wurde die Identifizierung des Fußgängers dahingehend gelöst, dass der Fußgänger als erkannt gilt, wenn er sich einen definierten Abstand hinter der Kante des Verdeckungsfahrzeuges befindet.
Das Modell unterscheidet bei der Erkennung des Fußgängers zwischen den drei Zuständen **not_detected**, **hidden** und **visible**. Der Zustand ergibt sich aus der berechneten Sichtbarkeit des Sichtbarkeitsmodells und der Sichtbarkeit aus dem Funktionsbock *FB_ Visibility_ Sensor*. Bei den Zuständen **hidden** und **visible** kann ein Fehler bei der Entfernungsmessung in x- und y-Richtung mit den Look-Up-Tabellen *kotag_ x/y_ error_ hidden* und *kotag_ x/y_ error_ visible* berücksichtigt werden. Bei verdecktem Fußgänger wird der Fehleraufschlag durch die Entfernung zur Fahrzeugkante definiert und bei voller optischer Sichtbarkeit durch die Entfernung zwischen Fußgänger und Sensor definiert.

Initialisierungsparameter:

```
%Init values for Ko-TAG_Model
distance_zero_point_front = 2.33;  %[m] Distanz Nullpunkt - Front
%ko-tag sensor
distance_kotag_zero_point = 2.33; %[m]  Position des Sensors zum Ego-Fahrzeug-Nullpunkt
kotag_distance = 70; %[m] Reichweite
kotag_angle = 80; %[°] Öffnungswinkel Ko-TAG 1 Seite
%Fehler voll sichtbar
load('kotag_x_error_visible.mat');
load('kotag_y_error_visible.mat');
%Fehler verdeckt
load('kotag_x_error_hidden.mat');
load('kotag_y_error_hidden.mat');
```

Ein- und Ausgangsgrößen:
Die Schnittstelle unterscheidet sich zum Kameramodell in der zusätzlichen Eingangsgröße *distance_ hidden* und im veränderten Wertebereiches der Ausgangsgröße *identified*.

```
distance_hidden %[m] Abstand zwischen Fußgänger und Verdeckungsfahrzeug
identified %[-] Sichtbarkeit: 0 - nicht sichtbar, 0.5 - verdeckt, 1 - sichtbar
```

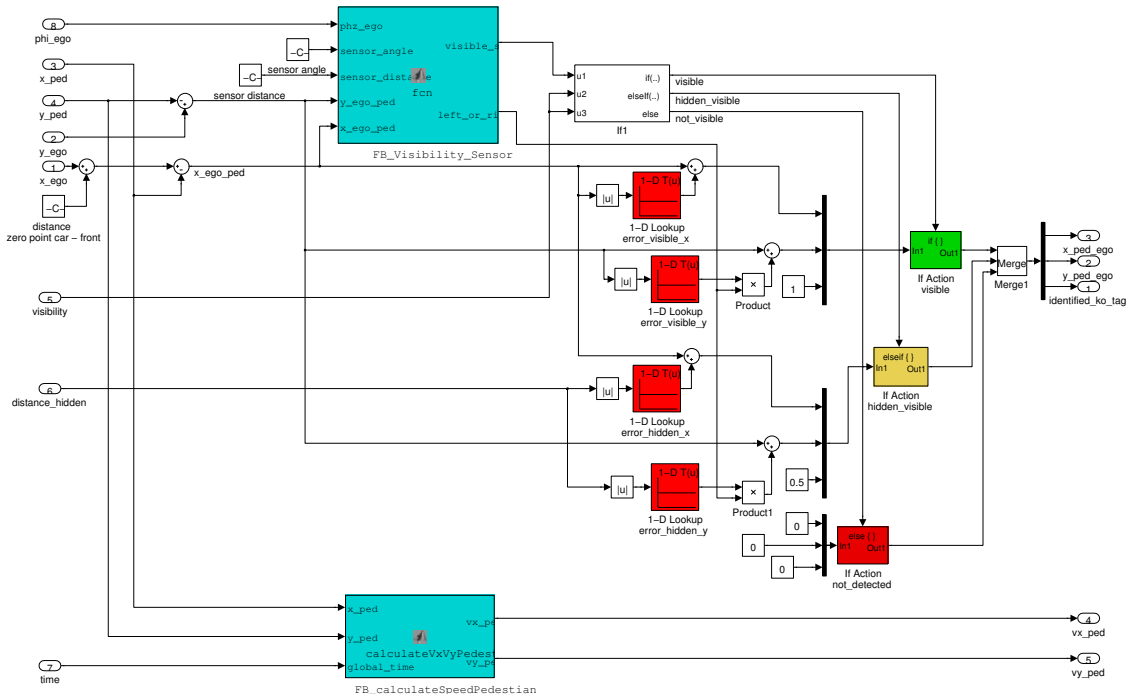

Abbildung 3.7: Implementierung von Ko-TAG in Matlab Simulink

3.5 Funktionsalgorithmus

Der Funktionsalgorithmus ist für die Analyse des Bereiches vor dem Fahrzeug beim präventiven Fußgängerschutz zuständig. Im Falle einer kritischen Situation, in der die Kollision nicht mehr vermeidbar ist, ist der Algorithmus für die Generierung der Warnung und die Einleitung einer automatischen Bremsung zuständig.
Der Funktionsalgorithmus teilt den Bereich vor dem Fahrzeug in drei geometrische Zonen, die durch die Time-to-Collison (TTC) und dem Abstand zum Fahrschlauch definiert sind. In Abbildung 3.8 sind die Zonen schematisch dargestellt. Diese sind durch eine feste seitliche Grenze und eine geschwindigkeitsabhängige Grenze in der Längsrichtung unterteilt. Je schneller sich das Fahrzeug bewegt, desto größer werden die drei Zonen in der Längsrichtung. Im Falle eines querenden Fußgängers wird der seitliche Grenze nach außen verschoben, um früher zu warnen oder einen nötigen Bremseingriff einzuleiten.

Befindet sich der Fußgänger im grünen Bereich, so erfolgt vom System noch keine Aktion, da eine Kollision vom Fahrer ohne Problem vermieden werden kann. Der Fußgänger wird vom System weiter beobachtet und seine Trajektorie wird bestimmt.
Sobald sich der Fußgänger in der Warnzone (gelben Bereich) befindet, bekommt der Fahrer eine akustische und optische Warnung, um auf die drohende Kollision aufmerksam zu

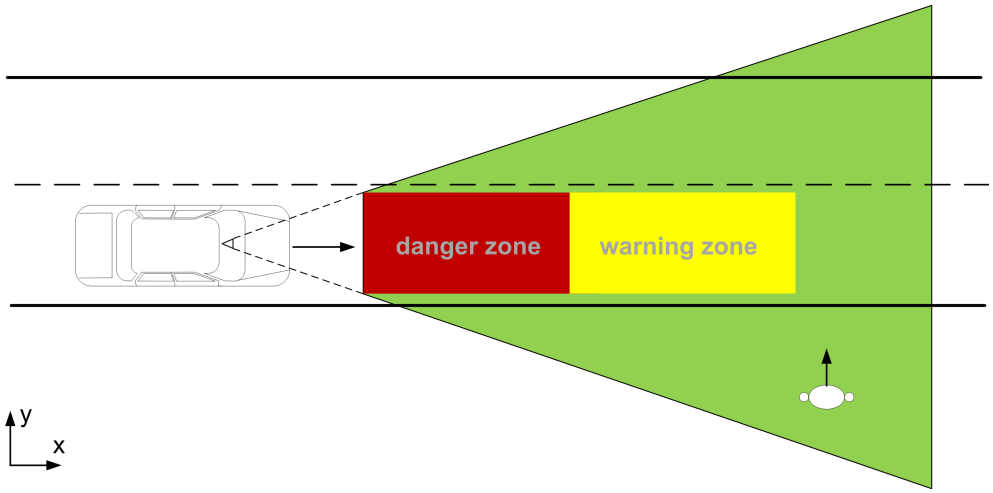

Abbildung 3.8: Schematische Darstellung des Warn- Bremskorridors

machen, da zu diesem Zeitpunkt die Kollision durch eine Reaktion des Fahrers vermieden werden kann. Des Weiteren erfolgt die Bremsvorkonditionierung, um ein schnelleres und stärkeres Ansprechen der Bremse zu erreichen. Aufgrund der fehlenden Möglichkeit beim verwendeten Versuchsfahrzeug wird nur eine Warnung ausgegeben.
Wenn sich der Fußgänger in der Gefahrenzone (rote Zone) befindet, steht eine Kollision des Fahrzeuges mit dem Fußgänger kurz bevor und der Funktionsalgorithmus kann eine autonome Bremsung einleiten. Dies bedeutet aber nicht, dass sobald sich der Fußgänger in der Gefahrenzone befindet, sofort gebremst wird. Da der Bremsweg geschwindigkeitsabhängig ist, kann die Einleitung bei niedrigen Geschwindigkeiten später erfolgen, um einerseits nicht unnötig früh zum Stehen zu kommen und andererseits um Falschauslösungen zu reduzieren.
Da das Ziel der Arbeit nicht die Entwicklung des Funktionsalgorithmus ist und dies den Rahmen der Arbeit um ein Vielfaches vergrößert hätte, wurde der Bremszeitpunkt geschwindigkeitsunabhängig auf einen fixen Zeitpunkt gelegt, um die Komplexität des Algorithmus zu reduzieren. Um die Simulation und die realen Fahrversuche zu vergleichen, wird in beiden Fällen der gleiche Algorithmus für den präventiven Fußgängerschutz verwendet. [Tie12]

Realisierung:
Der Funktionsalgorithmus für den präventiven Fußgängerschutz wurde in C++ entwickelt. Aufgrund der Verwendung des Algorithmus in der Simulation und in der realen Anwendung besteht der Vorteil, dass nur ein Algorithmus für beide Fälle entwickelt werden muss. Keine Portierung des Algorithmus in eine andere Sprache ist notwendig. In der Simulation wird ein Matlab Simulink Modell mit Hilfe des s-Function-Builder erstellt. Im Versuchs-

träger wird der Algorithmus im Micro Framework (Näheres siehe Kapitel 4.2.2 MicroFW) ausgeführt, das Framework wurde in C++ realisiert und eine Integration des Algorithmus ist somit effizient möglich.

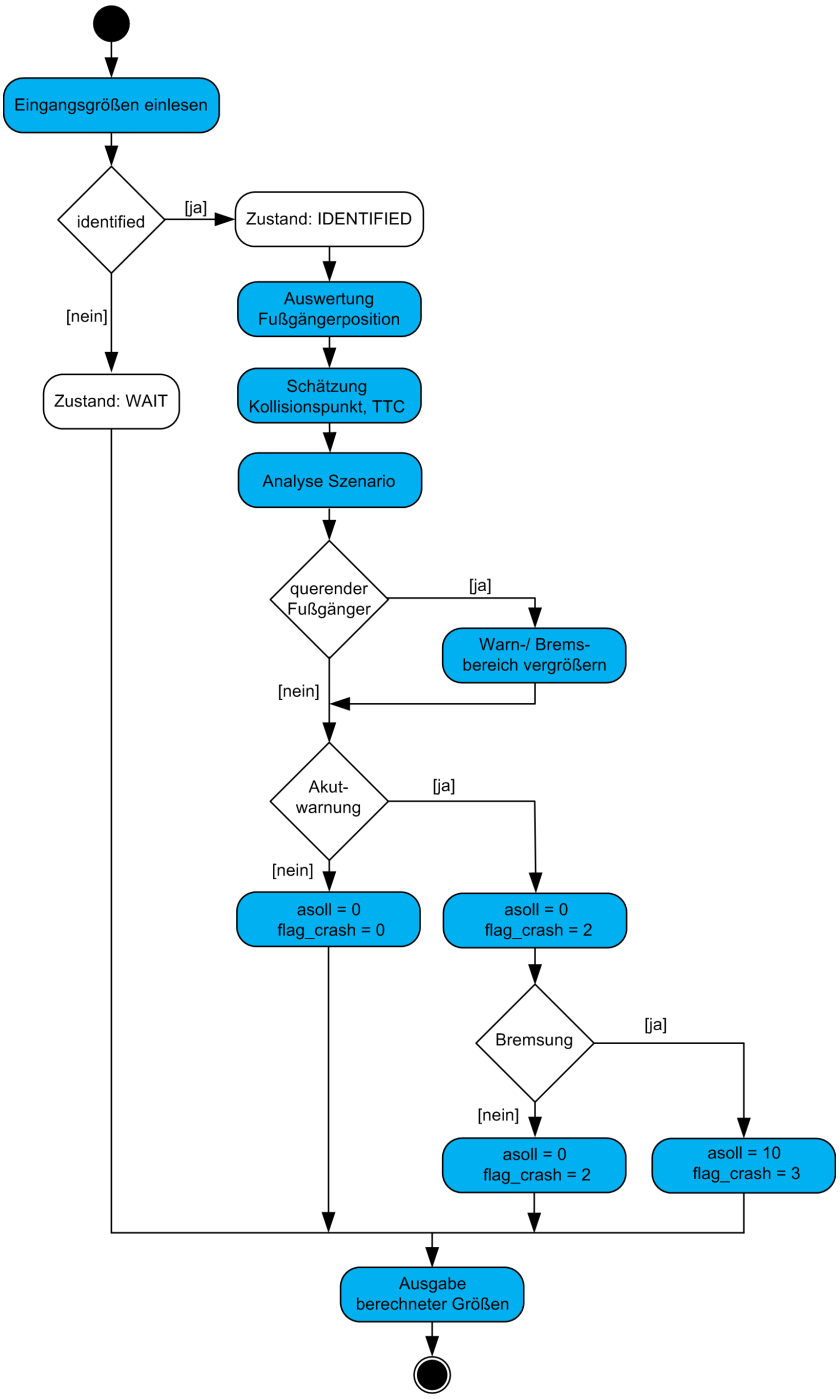

Abbildung 3.9: Grafischer Ablauf des Funktionsalgorithmus

Der Funktionsalgorithmus wurde als Zustandsmaschine mit zwei Zuständen realisiert: WAIT und IDENTIFIED. Solange der Sensor keinen Fußgänger in seinem Sichtfeld erkannt hat, bleibt der Algorithmus in diesem Zustand. Sobald der Sensor einen Fußgänger identifiziert hat (verdeckt oder sichtbar), wird in den Zustand IDENTIFIED gewechselt, in dem der Algorithmus die Sensordaten auswertet. Aus den Sensordaten und den Fahrzeugdaten werden die daraus resultierende TTC und die vermutliche Position des Fußgängers zum Zeitpunkt t0 berechnet. Aus dem Bewegungsverlauf des Fußgängers wird analysiert, ob es sich um einen kreuzenden Fußgänger handelt. Wenn sich ein Fußgänger in die Richtung des Fahrschlauches bewegt, so vergrößert sich der Warn- und Bremskorridor in y-Richtung. Anschließend erfolgt die Analyse, ob sich der Fußgänger bei der aktuellen Position bzw. zum Kollisionszeitpunkt in einem Warn- oder Bremskorridor befindet. Im letzten Schritt erfolgt eine Weiterleitung der Daten an das Steuergerät für die Längsdynamik, das für die Bremsung zuständig ist. Der Funktionsalgorithmus ist über folgend definierte Funktion ausführbar:

pFGS*(time, x_ped_ego, y_ped_ego, identified, vx_ego, vy_ego, vx_ped, vy_ped,*
**distance, * detect, * vdiff, *flag_crash,* asoll, *ttc, *collisionstatus,*
**debug_out1, *debug_out2, *debug_out3)*

Eingangsgrößen:

```
time %[s] global Zeit
x_ped_ego %[m] x-Abstand Ego-Fahrzeug Fußgänger
y_ped_ego %[m] y-Abstand Ego-Fahrzeug Fußgänger
identified %[-] Sichtbarkeit des Fußgängers: 0 - nicht sichtbar, 0.5 - verdeckt, 1 - sichtbar
vx_ego %[m/s] Geschwindigkeit des Fußgängers in x-Richtung
vy_ego %[m/s] Geschwindigkeit des Fußgängers in y-Richtung
```

Ausgangsgrößen:

```
distance %[m] Abstand in x-Richtung zwischen Ego-Fahrzeug und Fußgänger
detect %[-] Fußgänger identifiziert: 0 - nein, 1 - ja
vdiff %[m/s] Differenzgeschwindigkeit zwischen Ego-Fahrzeug und Fußgänger
flag_crash %[-] 0 - keine Gefahr, 2 - Vorwarnung, 3 - Akutwarnung
asoll %[m/(s^2)] Sollverzögerung Ego-Fahrzeug
ttc %[s] Time-to-Collision Ego-Fahrzeug Fußgänger
collisionstatus %[-] Kollision: 0 - nein, 1 - ja
```

Initialisierungsparameter:

```
%Init values for Function_Algorithm
update_rate = 0.07;  %[s]  Abtastrate des Funktionsalgorithmus
```

3.6 Modellierung Ego-Fahrzeug (Bremsvorgang)

Das Ego-Fahrzeug bewegt sich bei den Szenarien nur in Längsrichtung, aus diesem Grund wird nur die Längsdynamik des Fahrzeuges modelliert. Das Modell soll die aktuelle Position und Geschwindigkeit des Fahrzeuges liefern und auf eine Bremsanforderung reagieren. Messungen am Prüfstand haben gezeigt, dass sich das Fahrzeug bei Messungen am Prüfstand nicht mit konstanter Geschwindigkeit bewegt, sondern in den letzten Sekunden leicht verzögert, deshalb muss in der Simulation berücksichtig werden, dass das Fahrzeug mit einer Geschwindigkeit startet und konstant leicht verzögert (z. B. -0.1m/s^2), damit die Simulationen mit den Messungen verglichen werden können. In Abbildung 5.6 *Vergleich des Geschwindigkeitsverlaufes – Messung und Simulation* ist der Geschwindigkeitsverlauf des Ego-Fahrzeuges dargestellt, die Verzögerung vor der Vollbremsung ist gut sichtbar.

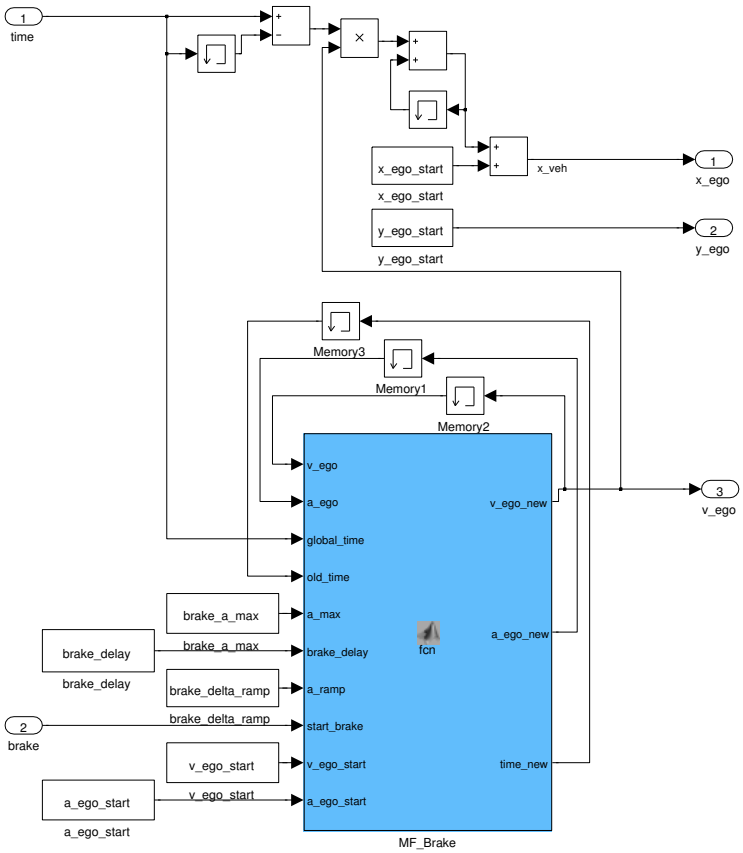

Abbildung 3.10: Implementation des Ego-Fahrzeuges in Matlab Simulink

Realisierung:

Die Modellierung des Bremsvorganges erfolgte in Matlab Simulink. Das Modell (Abbildung 3.10) liefert als Ausgangsgröße die aktuelle Position und Fahrzeuggeschwindigkeit. Über

die Eingangsgröße a_brake_soll lässt sich das Fahrzeug mit der gewünschten Verzögerung bremsen. Sobald eine Bremsung eingeleitet wurde, verzögert das Modell bis zum Stillstand des Fahrzeuges.
Der Funktionsblock *MF_Brake* ist für die Berechnung der Geschwindigkeit des Fahrzeuges zuständig. Abhängig von der letzten Geschwindigkeit, der Verzögerung und einer Bremsanforderung wird die aktuelle Geschwindigkeit berechnet und daraus die aktuelle Position des Fahrzeuges ermittelt.
Bei einer Bremsanforderung wird die Verzögerung konstant inkrementell erhöht, bis die maximal mögliche Verzögerung des Fahrzeuges erreicht wird, danach wird bis zum Stillstand die Verzögerung konstant aufrechterhalten. Die Dauer, bis die max. Verzögerung erreicht wird, ergibt sich aus der Steigung des Verzögerungsaufbaus und der max. Verzögerung.
Die Parameter für das Bremsmodell werden über Messungen am Prüfstand bestimmt, dazu mehr in Kapitel 5.3.1 Parametrisierung Bremsverlauf.

Initialisierungsparameter:

```
%Init values for Ego_Vehicle
x_ego_start = -18.83 % [m] Startposition Ego-Fahrzeug x-Richtung
y_ego_start = 0 % [m] Startposition Ego-Fahrzeug y-Richtung
v_ego_start = 8.1 % [m/s] Anfangsgeschwindigkeit Ego-Fahrzeug
a_ego_start = -0.1 % [m/s*s] Verzögerung Ego-Fahrzeug

brake_delay = 0.21 % [s] Totzeit des Bremssystems
brake_a_max = -8.1 % [m/s*s] maximale Verzögerung
brake_delta_ramp = 20 % a_max / delta_ramp = Dauer bis die max. Verzögerung aufgebaut wird
```

Eingangsgrößen:

```
Time % [s] Globale zeit
brake % 0 - keine Bremsung, 1 - Bremsung
```

Ausgangsgrößen:

```
x_ego % [m] x-Position Ego-Fahrzeug
y_ego % [m] y-Position Ego-Fahrzeug
phi_ego % [rad] Ausrichtungswinkel Ego-Fahrzeug
v_ego % [m/s] Geschwindigkeit Ego-Fahrzeug
```

3.7 Independent Co-Simulation - ICOS

Die Co-Simulationsplattform ICOS [Vif13] stellt für eine Vielzahl an Simulationstools (z. B. Abaqus, Adams, Cruise, Matlab Simulink, ...) definierte Schnittstellen für den Datenaustausch zur Verfügung. Über diese Schnittstellen kann ICOS den Datenaustausch zwischen den unterschiedlichen Simulationsumgebungen steuern. Je nach Simulationsumgebung können für den Datenaustausch die Variablen skalar oder als Vektor definiert sein und beim

Datentyp kann zwischen Integer, Double oder String gewählt werden. Einzige Voraussetzung für den Datenaustausch zwischen unterschiedlichen Modellen ist die Einhaltung dieser Bedingungen.
Anhand der Einbindung des Ko-TAG-Sensors in die Co-Simulation soll im folgenden Abschnitt näher auf die Integration des Matlab Simulink Modelles in ICOS eingegangen werden.

3.7.1 Adaptierung des Ko-TAG-basierten Sensormodelles in Simulink

Das Sensormodell wurde in Matlab Simulink entwickelt. ICOS stellt für Matlab Simulink einen Wrapper zur Verfügung, der die Möglichkeit bietet, Ein- und Ausgangsvariablen von Simulink an ICOS anzubinden. Bei jedem Zyklus stellt ICOS die Eingangsvariablen für Simulink zur Verfügung, die Ausgangsvariablen vom Modell werden ICOS zur Verfügung gestellt und dort können diese an andere Modelle weitergeleitet werden.
Die über den Library Browser in Simulink verfügbaren Funktionsblöcke *InPort* und *OutPort* von ICOS müssen in das Modell gezogen und mit einem Signal verbunden werden. Abbildung 3.11 stellt das Ko-TAG-basierte Sensormodell mit den InPort und OutPort Funktionsblöcken für die Anbindung an ICOS dar.

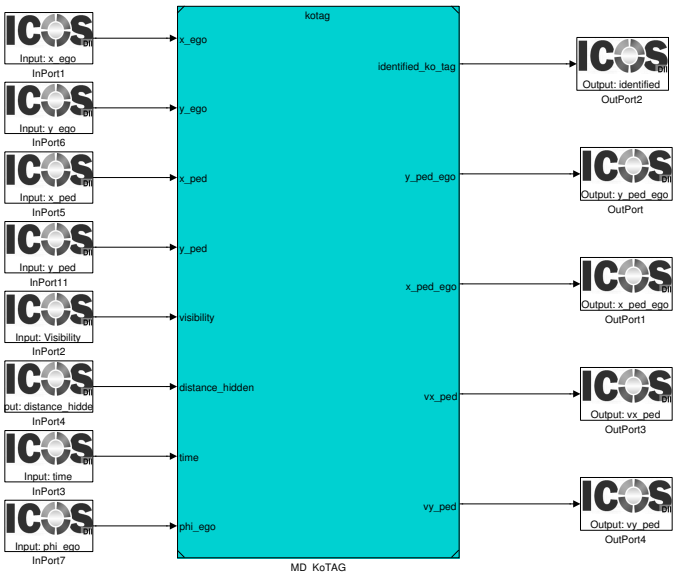

Abbildung 3.11: Ko-TAG-Sensormodell mit den InPorts und OutPorts für ICOS

Damit die Anbindung mit ICOS funktioniert, muss nur die Simulationszeit beim Solver eine variable Zykluszeit eingestellt werden, da über ICOS die Zykluszeit vorgegeben wird. Ansonsten müssen in Matlab keine weiteren Vorkehrungen bzw. Anpassungen am Modell durchgeführt werden.

3.7.2 Einbindung des Sensormodelles in ICOS

In Abbildung 3.12 wird der *Configuration Screen* der Simulationsumgebung ICOS dargestellt, im oberen Bereich befindet sich eine Liste mit allen beteiligten Modellen und im unteren Bereich bei *Model Setting* wird die gesamte Konfiguration des ausgewählten Modells aufgelistet.

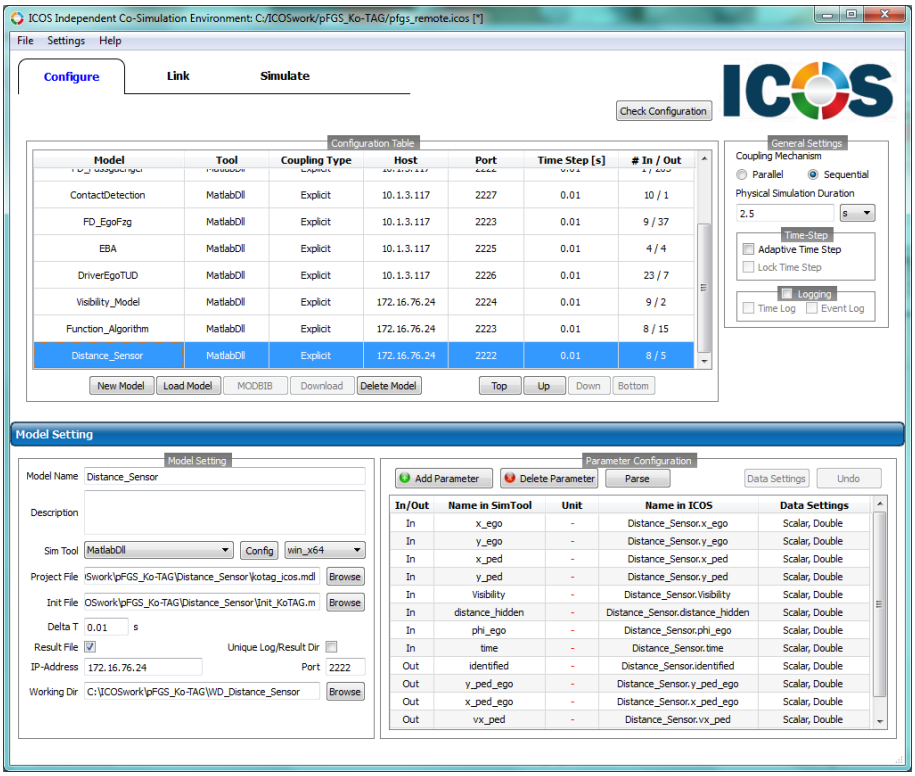

Abbildung 3.12: Übersicht der Simulationsumgebung ICOS

Folgende Parameter müssen eingestellt werden:

- Model Name: Distance_Sensor
- Sim Tool: z. B. MatlabDll
- Project File: Verzeichnis des Matlab Modelles
- Init File: optinal, z. B. für Initialisierungen von Variablen
- Delta T: Zykluszeit z. B. 0.01 s
- IP-Address: Rechner auf dem sich das Modell befindet und gerechnet wird
- Working Dir: Verzeichnis für das Simulationsergebnis

Die Ein- und Ausgangsvariablen des Modelles werden im Bereich *Parameter Configuration* dargestellt. Hier kann man auswählen, ob es sich um einen InPort oder OutPort handelt

und welcher Datentyp für den Datenaustausch verwendet wird. Bei der Konfiguration der einzelnen Variablen ist zu beachten, dass der gleiche Name wie in Simulink verwendet werden muss, damit ICOS die Variablen eindeutig zuordnen kann.

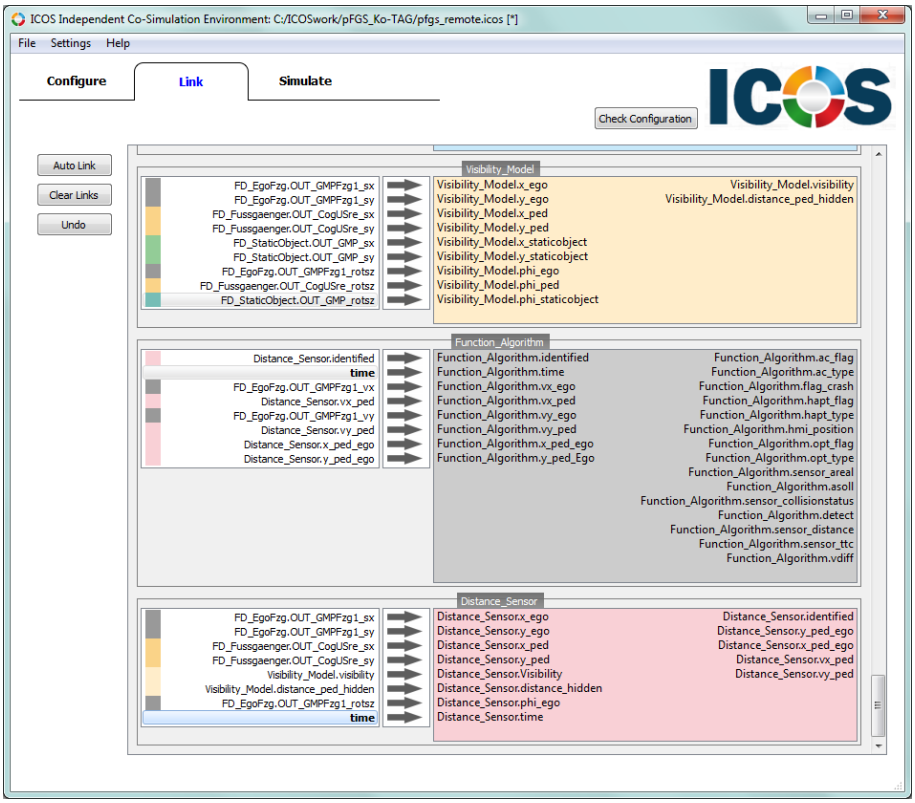

Abbildung 3.13: Link Screen der Simulationsumgebung ICOS

3.7.3 Verbinden der einzelnen Teilmodelle

Wenn alle benötigten Modelle in ICOS konfiguriert sind, können diese über den *Link Screen* miteinander verbunden werden. Im *Link Screen* sind alle Modelle die im *Configuration Screen* definiert sind, mit ihren Ein- uns Ausgangsvariablen dargestellt. Damit die Simulation gestartet werden kann, müssen jedem Modell alle Eingangsvariablen zugewiesen werden. Es besteht die Möglichkeit, dass Eingangsvariablen von einem anderen Modell oder vom **Boundary Condition Server** (BCS) zur Verfügung gestellt werden. Mit dem Boundary Condition Server ist es möglich eine Eingangsgröße als konstant oder über den zeitlichen Verlauf variabel zu definieren. In Abbildung 3.13 sind die drei Modelle *Visibility_ Model, Function_ Algorithm* und der *Distance_ Sensor* mit den Ein- und Ausgangsvariablen und den entsprechenden Eingangsgrößen von anderen Modellen dargestellt.

3.7.4 Verteilung der Modelle

ICOS bietet die Möglichkeit, die Simulation verteilt durchzuführen, einzelne Modelle können sich somit auf unterschiedlichen Rechnern befinden. Voraussetzung dafür ist eine gegenseitige Erreichbarkeit aller ICOS Remote Server, auf denen die Modelle ausgeführt werden. Damit der Datenaustausch zwischen den Servern funktioniert, muss für jeden ein Port verfügbar sein, über den die Kommunikation stattfindet. Die Konfiguration, auf welchem Server welches Modell ausgeführt werden soll, erfolgt im Configuration Screen.
Aus organisatorischen Gründen wurde die Simulation des präventiven Fußgängerschutzes verteilt durchgeführt. Dabei wurden die bereits bestehenden Modelle und die in dieser Arbeit entstandenen Modelle auf separaten Rechnern ausgeführt.

3.8 Simulationsergebnisse präventiver Fußgängerschutz

In Kapitel 5 wird die Anwendung der Co-Simulation, am Beispiel des präventiven Fußgängerschutzes, beschrieben. Zudem wird auf die Parametrisierung der Teilmodelle durch Messfahrten am Fußgängerprüfstand eingegangen. Die Anwendung der Co-Simulation des kamerabasierten und Ko-TAG-basierten Fußgängerschutzes wird in Punkt 5.3.4 skizziert. In diesem Punkt werden die Ergebnisse der Messungen am Prüfstand, die daraus gewonnenen Simulationsparameter und die Ergebnisse der durchgeführten Simulationen dargestellt.

Kapitel 4

Integration in das Fahrzeug

Für die Messfahrten zur Auswertung der kamerabasierten Fußgängererkennung und des Ko-TAG-Sensors sowie zur Funktionsentwicklung des Warn-/Bremsalgorithmus für den Ko-TAG-basierten präventiven Fußgängerschutz wurde für das Projekt ein BMW 550i Touring (F11) ausgerüstet. In diesem Kapitel werden die benötigten Komponenten kurz beschrieben und auf die Umbauarbeiten am Versuchsfahrzeug wird eingegangen.

Um mit dem Fahrzeug Messfahrten durchzuführen, wurde das Fahrzeug zu Beginn mit einer Grundausstattung an Komponenten ausgerüstet:

- Messrechner
- Energiesteuerung
- Switch
- WLAN-Router
- ZGW (Zentrales Gateway) zum Routen zwischen Bussen

Am Fahrzeug erfolgte die mechanische Integration des Ko-TAG-Sensors in die vordere Stoßstange. Für die Aufzeichnung der Messdaten vom Fahrzeug und der Sensorik wurden die benötigten Busleitungen in den Kofferraum verlegt bzw. nach Bedarf angepasst. Damit die einzelnen Komponenten gezielt ein- und ausgeschaltet werden können, werden diese von einer separaten Energiesteuerung versorgt, welche direkt an der Batterie angeschlossen ist. Einen Überblick über die eingebauten Komponenten und die Verkabelung gibt die Abbildung 4.1.

Kamerabasierte Fußgängererkennung
Für die kamerabasierte Fußgängererkennung gibt es bereits einen Vorserienstand. Hierbei wird mit der Kamera-ECU das Kamerabild ausgewertet und die Position des Fußgängers ermittelt. Aufgrund der Position des Fußgängers und der Fahrzeuggeschwindigkeit wird mit Hilfe eines Algorithmus und dem definierten Warn- und Bremskorridor analysiert, ob eine

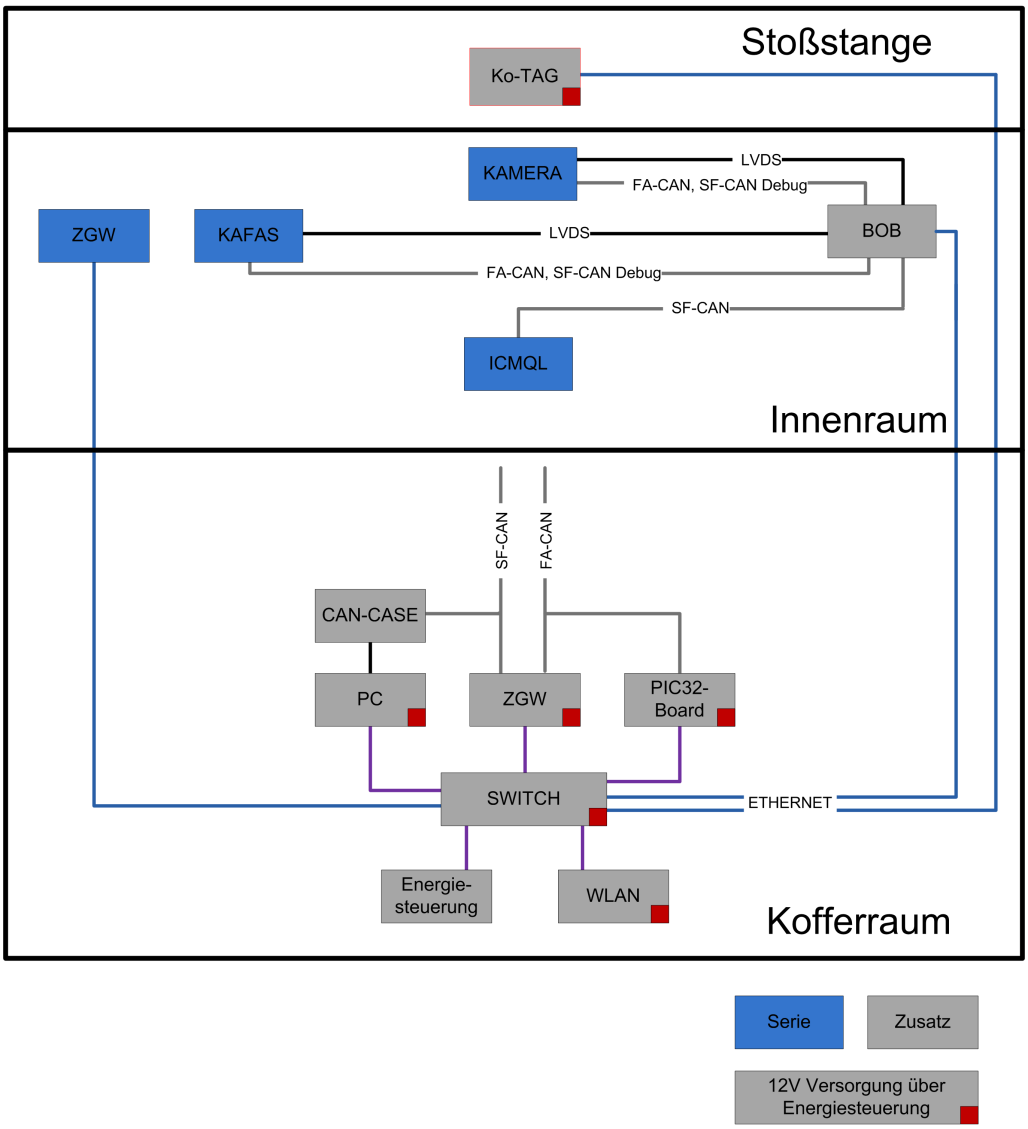

Abbildung 4.1: Übersicht aller Komponenten

Gefahrensituation vorliegt. Die Kamera-ECU generiert eine Warnung oder fordert einen Bremseingriff über den Bus an.

Um einen Bremseingriff durchzuführen, wird eine modifizierte Software auf dem Zentralen Gateway verwendet, mit dem es möglich ist durch Busmanipulation eine Bremsung einzuleiten. Die Bremsschnittstelle hat deshalb keinen Serienzustand und kann nicht mit der in der Serie verglichen werden. Für die Datenaufzeichnung wird das ADTF-Studio von Elektrobit verwendet [Eag13], dafür befindet sich ein Rechner im Kofferraum.

Ko-TAG-basierte Fußgängererkennung

Der kooperative Sensor Ko-TAG ist im Fahrzeug prototypisch integriert, die Analyse und Auswertung der Daten erfolgt auf einem Rechner im Kofferraum des Fahrzeuges. Über Ethernet ist der Sensor mit dem Rechner verbunden. Zur Bestimmung der Fahrzeugdaten wie Geschwindigkeit oder Gierrate wird ein CANcase von Vector verwendet.

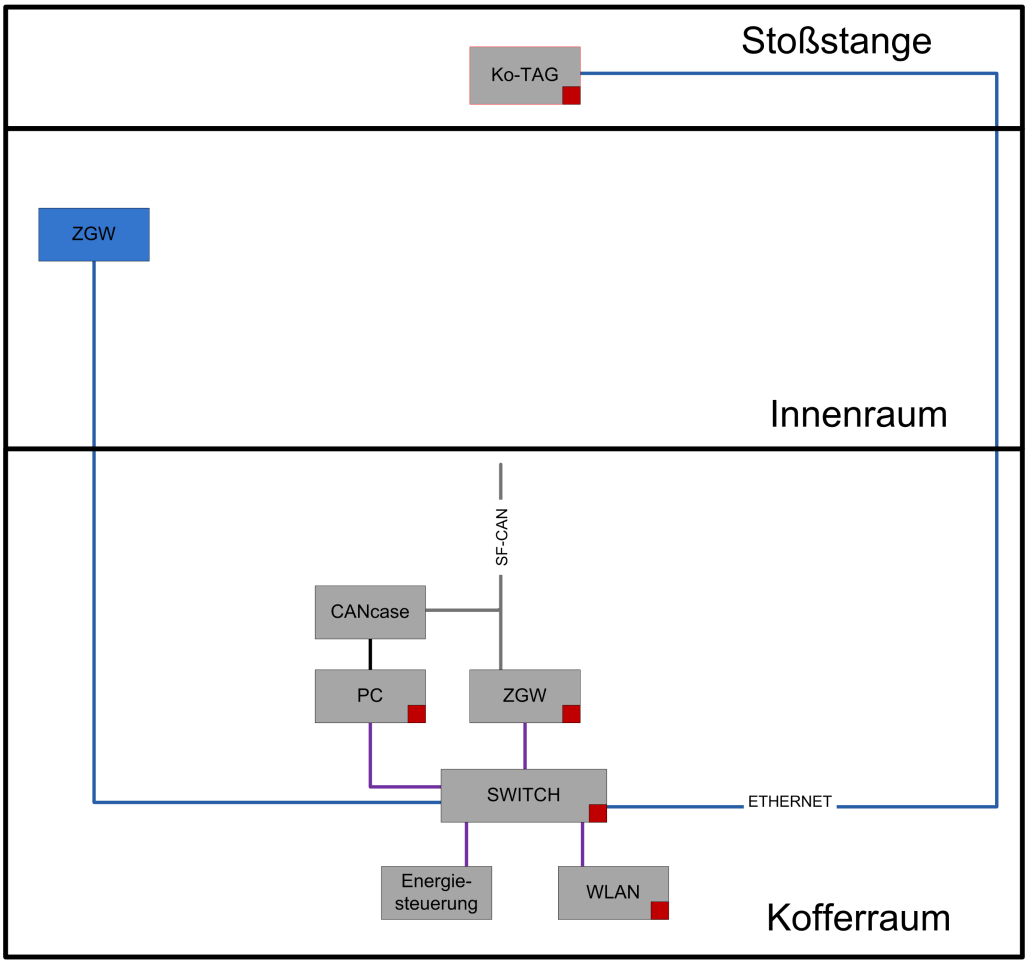

Abbildung 4.2: Ko-TAG-basierte Fußgängererkennung

Der Bremseingriff wird wie bei der oben beschriebenen kamerabasierten Fußgängererkennung durchgeführt. Um die Übersicht zu vereinfachen, werden in Abbildung 4.2 nur die Komponenten gezeigt, welche für einen Bremsung auf Fußgänger mit Ko-TAG benötigt werden.

4.1 Hardware Integration

4.1.1 Integration Ko-TAG

Damit der Ko-TAG-Sensor eine nahezu uneingeschränkte Kommunikationsverbindung nach vorne hat und von außen nicht sichtbar ist, befindet sich der Sensor direkt hinter der vorderen Stoßstange des Fahrzeuges. Am vorderen Querträger ist der Sensor durch eine speziell angefertigte Halterung befestigt. Die Modifikationen an der vorderen Stoßstange sowie die zusätzlich verbauten Komponenten im Kofferraum sind in Abbildung 4.3 dargestellt.

Die Kabel für die Spannungsversorgung (2-polig) und die Datenübertragung (CAT-5-Kabel) des Sensors sind entlang des vorderen rechten Kotflügels über den Unterboden und der Kofferraummulde in den Kofferraum verlegt. Das CAT-5-Kabel des Ko-TAG-Sensors ist mit dem Switch verbunden und die Spannungsversorgung mit der Energiesteuerung. Durch die Energiesteuerung kann der Sensor je nach Bedarf ein- oder ausgeschaltet werden.

Abbildung 4.3: Integration des Ko-TAG-Sensors in den Versuchsträger - oben: Ko-TAG-Sensor, unten links: Luftkühlung, unten rechts: Messtechnik im Kofferraum

Für den Einbau des Sensors in das Fahrzeug wurden folgende Modifikationen am Fahrzeug vorgenommen:

- Aufgrund der aktiven Luftkühlung des Sensors befindet sich auf der linken Seite vor dem Radkasten der Ansaugschlauch für Frischluft und ein 12V-Lüfter, der die Luft durch die Kühlrippen hinter dem Sensor auf die rechte Seite des Fahrzeuges bläst. Da das Fahrzeug mit einer Standheizung ausgestattet ist und sich die Standheizung, im Besonderen der Auspuff, im Bereich der Kühlung des Sensors befindet, wurde dieser entfernt. Ein kompletter Ausbau der Standheizung wurde nicht empfohlen, da die Standheizung mit dem Kühlkreislauf des Motors verbunden ist. Aus diesem Grund wurde nur die Kraftstoffpumpe der Standheizung abgesteckt.

- Durch die verbaute Sonderausstattung "Aktiver Fußgängerschutz" befindet sich direkt hinter der vorderen Stoßstange das Band, mit dem der Aufprall des Fußgängers detektiert wird. Durch die Integration von Ko-TAG muss dieses entfernt werden. Um eine Falschauslösung der aktiven Motorhaube auszuschließen, sind die vier Aktoren der Motorhaube abgeschlossen und die Ausgänge beim Airbag-Steuergerät auscodiert.

4.1.2 Integration Kamerabasierte Fußgängererkennung

Der Versuchsträger war noch mit der vorhergehenden Generation des Kamera-ECU ausgerüstet, mit welchem eine Fußgängererkennung noch nicht möglich ist. Deshalb wurde der Versuchsträger mit einem Vorseriensteuergerät ausgerüstet, das eine Fußgängererkennung unterstützt. Zum Aufzeichnen des Kamerabildes und der Objektdaten der Kamera wird eine Break-Out-Box verwendet, mit der alle es möglich ist mehrere CANs und das Kamerabild synchron aufzuzeichnen. Mit der Break-Out-Box ist auch möglich die das Kamerabild und Fahrzeugdaten im ADTF-Studio (Automotive Data and Time-Triggered Framework) zu visualisieren. Mehr Details zur Aufzeichnung und Visualisierung der Kameradaten ist im Kapitel 4.2.1 angeführt.

4.1.3 Zentrales Gateway

Das Seuergerät ZGW dient im Fahrzeug als Gateway zwischen den unterschiedlich verbauten Bussen, es routet Daten zwischen Flexray, Highspeed-CAN, Lowspeed-CAN und Ethernet.
Mit Hilfe eines zusätzlich verbauten ZGWs, mit speziellem Softwarestand ist es möglich Daten gezielt zwischen den CANs und Ethernet zu routen. Das ZGW lässt sich so konfigurieren, dass es Frames der einzelnen CANs über Ethernet weiterleitet bzw. über Ethernet empfangene UDP-Pakete auf den gewünschten CAN weiterleitet.
Zum Aufzeichnen der Daten vom Ko-TAG-Sensor werden die Daten im Messrechner ausge-

wertet und über Ethernet an das ZGW gesendet. Dieses routet die Daten auf den SF-CAN (Sensor-CAN) weiter, der für die Analyse der Messfahrten aufgezeichnet wird.

4.1.4 Rechner

Der Rechner im Fahrzeug erfüllt zwei Aufgaben. Einerseits dient er zum Aufzeichnen der Messdaten über das ADTF-Studio von Elektrobit und andererseits zur Anwendung des präventiven Fußgängerschutzes mit Ko-TAG. Im Kapitel Software-Integration wird näher auf die beiden angeführten Aufgaben eingegangen.

4.1.5 Energiesteuerung

Die Energiesteuerung (Abbildung 4.4) versorgt die zusätzlich verbauten Komponenten (Messtechnik, Sensorik, Steuergeräte, usw.) im Versuchsträger mit Strom, um einen reibungslosen Betrieb bei Messfahrten bzw. Erprobungsfahrten zu gewährleisten. Damit das Spannungssystem nicht überlastet wird, schaltet die Energiesteuerung Komponenten abhängig vom Fahrzeugzustand an oder aus, z. B. wenn die Batterien nicht durch die Lichtmaschine geladen wird. 20 Ausgänge können nach Fahrzeugzustand manuell oder automatisch konfiguriert werden. Die Energiesteuerung wurde im Laufe der Diplomarbeit dahingehend entwickelt, dass sie flexibel einsetzbar ist und für einen standardisierten Aufbau von Versuchsträgern geeignet ist.

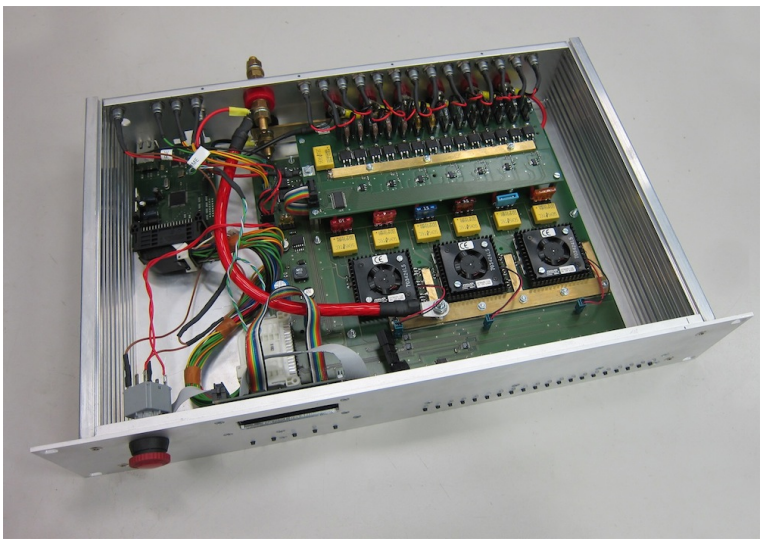

Abbildung 4.4: Energiesteuerung

Funktionsumfang:

- 14 x 5A Ausgänge (Lemo-Buchsen)
- 6 x 30A Ausgänge (Bananen-Buchsen 4mm)
- 3 x 12V Relais-Ausgänge
- Schnittstellen: 2 x CAN, 1 x Ethernet
- Funktionen: automatisches Aufwachen, gezieltes Herunterfahren, Überwachung der Ausgänge
- Manuelle Bedienung über die Gehäusefront oder die Windows-Applikation
- Automatischer Modus durch Konfiguration über die Windows- Applikation
- Display zur Status- und Fehleranzeige
- Steuerplatine: PIC32-Board
- Luftkühlung gesteuert über Temperaturüberwachung

Mittels einer Windows- Applikation erfolgt die Konfiguration der Ein- und Ausschaltzeitpunkte der Ausgänge für den automatischen Betrieb, zusätzlich können Ausgänge überwacht und manuell über Ethernet, CAN-Messages oder Taster an der Gehäusefront gesteuert werden. Jeder Ausgang kann so konfiguriert werden, dass er sich bei einem bestimmten Fahrzeugzustand (z. B. Zündung aus) nach einer vom Benutzer konfigurierten Zeit ein- oder ausschaltet. Um z. B. das Herunterfahren eines Messrechners zu ermöglichen, wird ständig ein Ethernet-Paket mit der Ausschaltzeit des zu abschaltenden Ausganges versendet. So kann die Komponente selbstständig herunterfahren und eventuell Daten rechtzeitig sichern, bevor die Trennung der Spannungsversorgung erfolgt.

Während des Betriebes werden alle Ausgänge ständig überwacht und im Falle eines Fehlers wird der entsprechende Ausgang deaktiviert. Über das Display an der Gehäusefront bzw. über die Applikation kann der Fehler abgerufen werden.

4.1.5.1 Hardware-Aufbau

In diesem Abschnitt wird kurz auf die einzelnen Platinen der Energiesteuerung eingegangen. Die Energiesteuerung wurde entwickelt, um folgende Ausbaustufen zu realisieren:

- Starkstromausgänge (30A) und Schwachstromausgänge (5A) mit oder ohne Bedienelement
- Starkstromausgänge mit oder ohne Bedienelement
- Schwachstromausgänge mit oder ohne Bedienelement

Um die unterschiedlichen Varianten zu realisieren, wurde die Energiesteuerung dahingehend entwickelt, dass sie aus unterschiedlichen Platinen besteht und je nach Bedarf zusammengebaut werden kann. Im Folgenden werden die einzelnen Platinen näher beschrieben.

Hauptplatine:
Bei der Energiesteuerung mit dem vollen Funktionsumfang wird die Hauptplatine benötigt. Über diese werden alle anderen Platinen, wie das PIC32-Board zur Steuerung der Energiesteuerung, die Adapterplatine für die Schwachstromausgänge und das Bedienelement, angeschlossen. Auf der Hauptplatine befinden sich die Starkstromausgänge (30A), die Spannungsversorgungen mit 3,3V und 5V und die Ansteuerung der drei Relais-Ausgänge. Bei jedem Stromausgang wird der Strompfad durch drei p-Kanal MOSFETs (parallel) geschaltet, um die Verlustleistung zu minimieren. Trotzdem werden die MOSFETs noch über einen Kühlkörper und kleine 12V-Lüfter gekühlt, damit es zu keiner Beschädigung der Bauteile kommen kann. Um eine Überlastung festzustellen, werden bei jedem der sechs Ausgänge die Ausgangsspannung, der Strom und die Temperatur der MOSFETs gemessen.

Adapterplatine:
Die Adapterplatine ist für die Schwachstromausgänge (5A) zuständig. Über I2C kommuniziert das PIC32-Board mit der Adapterplatine, auf ihr befindet sich ein MAX7300 28-Port I/O Expander, der die 14 p-Kanal MOSFETs steuert und die Ausgangsspannung der einzelnen Ausgänge über einen Spannungsteiler misst. Bei einer Abfrage des PIC32-Board über I2C leitet dieser die Werte weiter, so wird überprüft, ob bei einem Ausgang 12V anliegen oder vielleicht die Sicherung durchgebrannt ist.

Bedienelement:
Um die Energiesteuerung manuell über die Gehäusefront zu steuern, wird das optionale Bedienelement benötigt. Das Bedienelement kann entweder direkt beim PIC32-Board oder bei der Hauptplatine angesteckt werden. Auf der Platine befindet sich ein PIC32 zur Auswertung aller verbauten Taster und zur Steuerung der LEDs für die Zustandsanzeige. Die Kommunikation des Bedienelements mit dem PIC32-Board erfolgt über die Serielle Schnittstelle.

PIC32-Board:
Für die Steuerung der Energiesteuerung wird das PIC32-Board verwendet, es ist für die Kommunikation über Ethernet und CAN, für Eingaben vom Bedienelement und für die Steuerung und Überwachung jedes einzelnen Ausganges zuständig. Nähere Informationen zum PIC32-Board sind im Kapitel PIC32-Board angeführt.

4.1.5.2 Konfiguration

Über eine Windows-Applikation kann die Energiesteuerung für den automatischen Modus konfiguriert werden, aber auch die manuelle Steuerung der einzelnen Ausgänge ist damit

möglich. Die Applikation wurde so gestaltet, dass der gesamte Datenaustausch über Ethernet erfolgt. Somit kann über einen PC, der mit dem Netzwerk im Fahrzeug verbunden ist, die Energiesteuerung bedient werden. Die Oberfläche wurde so gestaltet, dass sie mit der Auflösung des Multimedia-Bildschirmes in der Mittelkonsole des Versuchsfahrzeuges übereinstimmt und bei Bedarf über einen Videokonverter dargestellt werden kann.
Die Ausgänge können nach folgenenden Fahrzeugzuständen programmiert werden: Sleep, Bus Activity, KL30B, KL15, Engine On und Driving. Bei einem Wechsel des Fahrzeugzustandes können nach einer optionalen Zeit die Ausgänge geschaltet werden.

4.2 Software Integration

4.2.1 Automotive Data and Time-Triggered Framework

Für die Aufzeichnung und Analyse der Kameradaten wurde das ADTF-Studio von Elektrobit verwendet. Es findet Anwendung bei der Entwicklung von Fahrerassistenzsystemen wie z. B. Nightvision, Verkehrsschilderkennung, Spurhalteassistent. Mit dem Tool kann das Kamerabild und relevante Busdaten aufgezeichnet werden. Zusätzliche Messdaten können grafisch dargestellt werden und im Nachhinein ist es möglich, die Messfahrten anzusehen und dabei Änderungen bei der Darstellung von Messwerten durchzuführen. Damit ist die Auswertung und Visualisierung der Messwerte sehr flexibel. Aus diesem Grund wurde das ADTF-Studio gewählt, da somit auch Messdaten vom Ko-TAG-Sensor dargestellt werden können.

Durch Overlays ist es möglich zusätzliche Messwerte am Videobild grafisch darzustellen wie z. B. die Positionsdaten des Fußgängers vom Prüfstand oder die aktuelle Fahrzeuggeschwindigkeit, siehe Abbildung 4.5. Dazu greift das ADTF-Studio auf die aufgezeichneten CAN-Nachrichten zurück und visualisiert die gewünschten Messwerte.
Um einen direkten Vergleich zwischen der Kamera und Ko-TAG zu ermöglichen, wurden die Messdaten von Ko-TAG auf den SF-CAN (Sensor-CAN) gelegt und vom ADTF-Studio aufgezeichnet. Für die Visualisierung dieser Messdaten werden zusätzliche Overlays benötigt. Für die Darstellung eines zusätzlichen Messwertes muss in den beiden unten angeführten xml-Dateien der gewünschte Messwert hinzugefügt werden. Als Beispiel ist hierfür die Darstellung der aktuelle x-Position des Fußgängers (koTAG_cs_x_ped) angeführt.

KAFAS_HIGH.PED_Overlay_CAN_config.xml
In der CAN_config-Datei werden alle Signale definiert, die vom ADTF-Studio visualisiert werden. Jedes Signal muss durch einen eindeutigen Namen definiert werden. Mit Hilfe des dbc-Files (data base CAN) und des darin gespeicherten Frame- und Signalnamen erfolgt die Verknüpfung. **KAFAS_HIGH.PED_Overlay_AVI_DISPLAY_config.xml**

Abbildung 4.5: ADTF-Aufzeichnung, Messfahrt am Fußgängerprüfstand mit Ko-TAG, Kamerabild des Fahrzeuges und Ko-TAG Overlay am linken Bildrand

Die Konfiguration der Darstellung eines Bildes im Kamerabild erfolgt über die AVI_DISPLAY_config-Datei. Hier wird festgelegt in welchem Bereich des Bildes und wie der Messwert dargestellt werden soll. Falls eine Umrechnung nötig ist, kann ein Offset bzw. ein Faktor definiert werden. Folgender Auszug zeigt die Darstellung der aktuellen x-Position des Fußgängers gemessen durch den Ko-TAG-Sensor:

```
<object name="KOTAG_CS_X" >
  <setting name ="show" value="false" />
  <setting name ="xPos" value="20" />
  <setting name ="yPos" value="260" />
  <!-- dataType "integer" or "float" or "string" or "static" -->
  <setting name ="dataType" value="float" />
  <setting name ="displayType" value="text" />
  <setting name ="defaultVal0" value="0" />
  <!-- printedValue = actualVal * factor + offset -->
  <setting name ="offset" value="0" />
  <setting name ="factor" value="1" />
  <setting name ="format" value="CS_X: %.2fm" />
  <setting name ="size" value="12" />
  <setting name ="font" value="Lucida Console" />
  <setting name ="color" value="0xFFFFFF" />
</object>
```

4.2.2 Micro Framework

Für die Aufzeichnung und Auswertung der Daten vom Ko-TAG-Sensor wird das MicroFW der BMW Forschung und Technik GmbH verwendet. Das Framework bietet die Möglichkeit, Daten unterschiedlichster Sensoren und Busdaten aufzuzeichnen, auszuwerten und gegebenenfalls Aktionen einzuleiten. Die Integration des Ko-TAG-Sensors in das Framework, die Positionsbestimmung und die Trajektorienberechnung des Transponders wurde bereits im Rahmen der Dissertation [Kld13] durchgeführt. Um aus den verrauschten Messwerten die Position bestmöglich zu berechnen, wird ein Extended-Kalman-Filter für die Schätzung der Position verwendet. Vom Framework werden die relativen Entfernungen und Geschwindigkeiten in x- und y- Richtungen des Transponders zur Verfügung gestellt und können im Framework weiterverwendet werden.

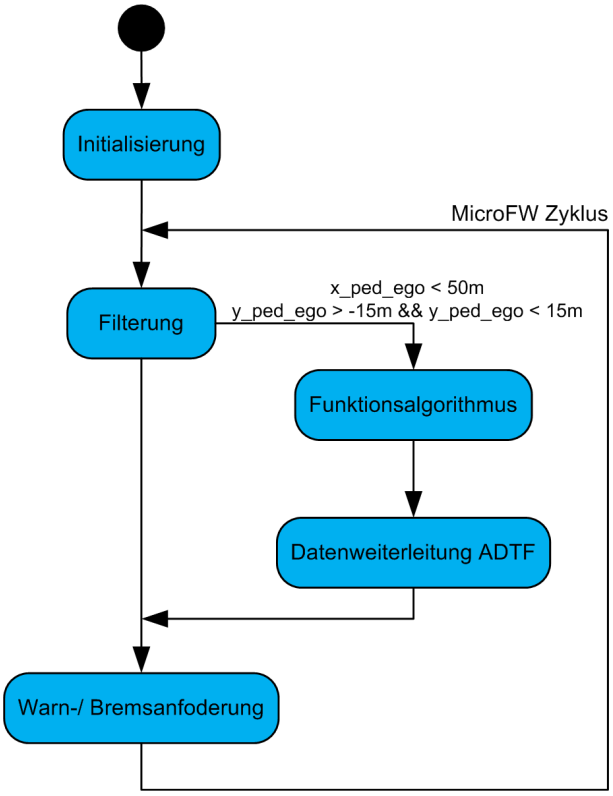

Abbildung 4.6: Funktionsablauf präventiver Fußgängerschutz

Integration Ko-TAG:
In Abbildung 4.6 ist die Integration des präventiven Fußgängerschutzes in das MicroFW skizziert. Das MicroFW hat eine Zykluszeit von 50 ms, innerhalb dieser Zeit erfolgen die Positions- und die Geschwindigkeitsbestimmung des Fußgängers, basierend auf den Sensordaten und den Fahrzeugdaten.

Der **Funktionsalgorithmus** und die **Datenweiterleitung ins ADTF** werden nur ausgeführt, wenn der Fußgänger sich in einem definierten Bereich vor dem Fahrzeug befindet **(Filterung)**. Wenn das Framework läuft wird die Warn-/ Bremsanforderung ständig gesendet, um die **Warn- bzw. Bremsanforderung** zu deaktivieren, wenn keine Positionsdaten vom Sensor kommen.
In den folgenden drei Unterkapiteln wird auf die Integration des Funktionsalgorithmus, der Bremsschnittstelle und der Datenweiterleitung für die Aufzeichnung der Daten im ADFT-Studio eingegangen.

4.2.2.1 Integration Funktionsalgorithmus Ko-Tag

Für den Funktionsalgorithmus im Fahrzeug wird derselbe Code wie in der Simulation verwendet, genauere Details dafür befinden sich in Kapitel 3.5. Im MicroFW erfolgt der Aufruf über die Methode ***pFGS****(time, x_ped_ego, y_ped_ego, identified, vx_ego, vy_ego, vx_ped, vy_ped, *distance, * detect, * vdiff, *flag_crash,* asoll, *ttc, *collisionstatus, *debug_out1, *debug_out2, *debug_out3)*, wenn vom Ko-TAG-Sensor gültige Positionsdaten geliefert werden und der Fußgänger weniger als 6m links/rechts in y-Richtung und 50m in x-Richtung vom Fahrzeug liegt.

4.2.2.2 Integration Datenweiterleitung über Ethernet

Um die Daten des Sensors und des Funktionsalgorithmus im ADTF-Studio aufzuzeichnen, werden die Daten vom MicroFW über Ethernet an das Gateway im Kofferraum weitergeleitet. Dieses routet die Frames auf den SF-CAN, wo die Daten mit der Breakout-Box aufgezeichnet werden. Mit der Methode ***sendPositionPedestrian****(pedestrian, can_frame_id)*, ***sendSpeedPedestrian****(pedestrian, can_frame_id)* und ***sendTTCPedestrian****(global_time, ttc, can_frame_id)* werden die Positionen des Fußgängers zu unterschiedlichen Zeiten, seine Geschwindigkeit und die TTC gesendet. Die in der Tabelle 4.1 angeführten Frames werden an den SF-CAN gesendet. In der Tabelle sind die CAN-Frames mit den dazugehörigen Signalen (Beschreibung, Startbit, Länge, Multiplikator, Offset, Einheit) angeführt.

4.2.2.3 Integration Bremsschnittstelle

Durch die definierte Schnittstelle beim Zentralen Gateway ist es möglich eine optische oder optisch-akustische Warnung darzustellen und eine Bremsung einzuleiten. Sobald das Mi-

Frame	Signal	Beschreibung	Startbit	Länge	Multiplikator	Offset	Einheit
0x7F4 (8 Byte)	KO_TAG_CS - aktueller Abstand zwischen Fahrzeug und Fußgänger						
	koTAG_cs_time	globale Zeit	0	32Bit	1	0	[ms]
	koTAG_cs_x_ped	x-Abstand	32	16Bit	0.01	-100	[m]
	koTAG_cs_y_ped	y-Abstand	48	16Bit	0.01	-100	[m]
0x7F5 (8 Byte)	KO_TAG_PS - geschätzter Abstand zwischen Fahrzeug und Fußgänger bei TTC = 0s						
	koTAG_ps_time	globale Zeit	0	32Bit	1	0	[ms]
	koTAG_ps_x_ped	x-Abstand	32	16Bit	0.01	-100	[m]
	koTAG_ps_y_ped	y-Abstand	48	16Bit	0.01	-100	[m]
0x7F7 (8 Byte)	KO_TAG_CS_TTC, Time-To-Collision						
	koTAG_cs_ttc_time	globale Zeit	0	32Bit	1	0	[ms]
	koTAG_cs_ttc	Time-To-Collision	32	16Bit	0.01	0	[s]
0x7F8 (8 Byte)	KO_TAG_DATEN_ALGO, Parameter des Funktionsalgorithmus						
	koTAG _algo_status	Warn-Bremsstatus	8	8Bit	1	0	[-]
	koTAG _algo_asoll	Sollverzögerung	24	8Bit	1	0	[m/s²]
	koTAG _algo_time	globale Zeit	32	32Bit	1	0	[ms]
0x7F9 (8 Byte)	KO_TAG_SPEED, aktuelle Geschwindigkeit des Fußgängers						
	koTAG_ps_time	globale Zeit	0	32Bit	1	0	[ms]
	koTAG_ps_vx_ped	Geschwindigkeit x	32	16Bit	0.01	-100	[m/s]
	koTAG_ps_vy_ped	Geschwindigkeit y	48	16Bit	0.01	-100	[m/s]

Tabelle 4.1: Gesendete Frames vom MicroFW

croFW am Rechner gestartet wird, sendet dieses zyklisch über Ethernet an das Zentrale Gateway die Nachricht mit der Warn- und Bremsanforderung, die durch den Funktionsalgorithmus berechnet wird. Dafür wird die Methode ***sendWarnBrakeRequest****(global_time, warn_brake_status, a, frame_id)* verwendet.

Die Nachricht wird nicht nur an das Zentrale Gateway vom Fahrzeug gesendet, sondern auch an das Gateway im Kofferraum, um den Warn- und Bremszeitpunkt im ADTF-Studio darzustellen. Dafür erfolgt der Methodenaufruf zweimal, jedoch unterscheidet sich die FrameID:

0x123 - Zentrale Gateway für den Bremseingriff

0x7F8 - Zentrale Gateway für Datenweiterleitung an das ADTF-Studio

4.2.3 Bremsschnittstelle

Über einen modifizierten Softwarestand am ZGW ist es möglich eine Bremsung einzuleiten. Mit einer UDP-Nachricht über Ethernet kann die Bremsung getriggert werden. Zusätzlich ist es möglich, über Ethernet die Vor- und Akutwarnung für den Fußgänger- und den Auffahrschutz auszulösen. Beim Softwarestand des Versuchsträgers ist es nicht möglich, die Bremse vorzukonditionieren bzw. die Bremsbeläge an die Bremsscheibe anzulegen, um für eine folgende Bremsung die Reaktionszeit zu verkürzen

4.3 Kommunikation Prüfstand – Fahrzeug

Der Prüfstand versendet zyklisch über WLAN die gemessenen Daten an das Fahrzeug. Kenngrößen wie die Position und Geschwindigkeit des Fußgängers und des Fahrzeuges sowie die Time-to-Collision werden übertragen. Mit Hilfe eines WLAN-Router werden die UDP-Pakete vom Prüfstand empfangen und mit einem PIC32-Board auf den FA-CAN des Fahrzeuges geroutet. Dadurch ist es möglich, die Daten des Prüfstandes im Fahrzeug durch das ADTF-Studio aufzuzeichnen, grafisch darzustellen und in weiterer Folge auszuwerten.

4.3.1 PIC32-Board

Für unterschiedlichste Anwendungsfälle bei der Konzeptintegration bzw. für die Absicherung von Komponenten wird abteilungsintern ein Mikrocontroller-Board mit dem PIC32 von Microchip verwendet [Mic13]. Das PIC32-Board verfügt über die typischen Mikrocontroller-Funktionen und für die Kommunikation mit anderen Komponenten im Fahrzeug stehen LIN, CAN und Ethernet zur Verfügung. Dadurch kann die Entwicklung von Anwendungen relativ schnell und einfach erfolgen. Für Anwendungen wie zum Beispiel Busmanipulationen, Routen zwischen unterschiedlichen Bussen oder Ansteuerung von Hardwarekomponenten kann das PIC32-Board verwendet werden.
Bei diesem Anwendungsfall erfolgt das Routen der UDP-Pakete des Prüfstandes auf den FA-CAN des Fahrzeuges.

Das PIC32-Board wurde parallel zur Diplomarbeit weiterentwickelt und dahingehend modifiziert, dass das PIC32 Ethernet Starter Kit von Microchip als Steuerplatine verwendet wird und eine Adapterplatine ansteuert, auf der die Peripherie (CAN, LIN, etc.) aufgebaut ist. Dadurch ist es möglich, den Funktionsumfang des PIC32-Boards zu ändern, ohne dass die gesamte Platine neu entwickelt werden muss. Werden für eine Anwendung z. B. mehrere 12V Ein-/Ausgänge oder ein GSM-Modem benötigt, wird nur die Adapterplatine neu entwickelt und aufgebaut, die Steuerplatine bleibt dabei unangetastet.
Der Arbeitsaufwand beim Aufbau einer Platine verringert sich dadurch wesentlich und der Großteil der Bauteile wird wieder verwendet.

4.3.1.1 Hardware

Funktionsumfang PIC32 Ethernet Starter Kit:

- 80MHz mit 512KB Flash (Programm + Daten), 128KB RAM
- ETHERNET
- USB Host Interface
- USB Program/Debug Interface
- C/C++ Compiler

Funktionsumfang Adapterplatine:

- 12V (KFZ-abgesichert)
- 2 x CAN (low- oder high-speed)
- 2 x LIN (2xSlave, 2xMaster oder 1xMaster und 1xSlave)
- 12V Relais Ausgänge
- 2 x I2C
- 1 x UART
- Digitale und analoge Ein-/Ausgänge

4.3.1.2 Software

Auch eine Basis-Software für das PIC32-Board wurde entwickelt, damit Projekte ohne Wissen über Buszugriffe bzw. richtiges Einstellen des Mikrocontrollers für Personen ohne große Programmierkenntnisse erstellt werden können.

Funktionsumfang Software:

- Teilapplikationen laufen in Tasks
- Schnittstellen: Ethernet-, CAN- und LIN-Stack, automatische Konfiguration der Interfaces
- Gateway Architektur zum Routen zwischen den Bussen (Weiterleitungs- und Manipulationsmöglichkeiten)

```
else if (identifier == RX_GW_ETH0_BODY_CAN_MSG_RT_BLT_CT_SOCCU_296_EthernetID)
{
memcpy(RX_GW_ETH0_BODY_CAN_MSG_RT_BLT_CT_SOCCU_296_BUFFER , aui8Payload,
       RX_GW_ETH0_BODY_CAN_MSG_RT_BLT_CT_SOCCU_296_Bytelength);
RX_GW_ETH0_BODY_CAN_MSG_RT_BLT_CT_SOCCU_296_RECEIVETIMER = time;
// Routing von ETH0 auf BODY-CAN
memcpy(TX_GW_CAN_BODY_CAN_MSG_ST_BLT_CT_KERT_291_BUFFER , aui8Payload,
      TX_GW_CAN_BODY_CAN_MSG_ST_BLT_CT_KERT_291_Bytelength);
}
```

- Einfaches Lesen/Schreiben der Daten am Bus, direkter Zugriff auf Signale in den Frames

  Auslesen eines Signals:

  ```
  RX_GW_ETH0_FA_CAN_MSG_RCOG_TR_345_RCOG_TR_SIGNAL_PRES_CLAS_LIM_V_GetSig(geschwindigkeit);
  ```

  Schreiben eines Signals:

  ```
  TX_GW_ETH0_FA_CAN_MSG_DISP_LDM_1_123_DISP_LDM_1_SIGNAL_DISP_LDM_CTRL_INFO_SetSig(ldm_m);
  ```

- Direktes Umwandeln physikalischer Größen:

  ```
  Value_phys = RX_GW_FA_CAN_MSG_G_VEH_321_G_VEH_SIGNAL_G_VEH_CEG_HexToPhys(hex);
  TX_GW_FA_CAN_MSG_V_TER_231_V_TER_SIGNAL_V_TER_COG_PhysToHex(phys)
  ```

4.4 Zusammenfassung

In Kapitel 4 wird die Integration des Ko-TAG-Sensors und der nötigen Messtechnik zur Datenaufzeichnung am Prüfstand erläutert. Damit das Fahrzeug auf Fußgänger bremst, werden die durchgeführten Umbauten am Fahrzeug und die Inbetriebnahme des Ko-TAG-Sensors im Detail beschrieben. Durch diese Modifikationen konnten die Messfahrten am Fußgängerprüfstand durchgeführt werden. Die Durchführung, Auswertung und Simulation ist im folgenden Kapitel angeführt.

Kapitel 5

Anwendung präventiver Fußgängerschutz

In diesem Kapitel sind alle Ergebnisse der Messungen am Fußgängerprüfstand und der Simulation angeführt. Bei der Vermessung der beiden Sensoren und der Erprobung des Funktionsalgorithmus des Ko-TAG-basierten präventiven Fußgängerschutzes wurden die in Tabelle 5.1 angeführten Szenarien durchgeführt. Die Definition der Szenarien stammt aus dem ersten Entwurf des vFSS Konsortiums (Advanced Forward Looking Safety Systems). Bei den Szenarien bewegt sich der Fußgänger immer vom rechten Fahrbahnrand in Richtung des Ego-Fahrzeuges. Der Kollisionspunkt, die Geschwindigkeit und die Verdeckung des Fußgängers variieren dabei. Bei den durchgeführten Messungen bewegt sich das Fahrzeug mit 30 km/h in Richtung des Kollisionspunktes mit dem Fußgänger, die Geschwindigkeit des Ego-Fahrzeuges wird nicht variiert.

	Fahrzeuggeschw. [km/h]	Fußgängergeschw. [km/h]	Kollisionspunkt [mm]	Verdeckung
Szenario 1	30	5	0	Ja
Szenario 2	30	8	470 (75%)	Ja
Szenario 3	30	8	-470 (25%)	Ja
Szenario 4	30	5	0	Nein
Szenario 5	30	3	0	Nein

Tabelle 5.1: Szenarien für den Fußgängerschutz

5.1 Versuchsaufbau

Für die durchgeführten Messungen ist der Versuchsaufbau in Abbildung 5.1 dargestellt. Das Verdeckungsfahrzeug befindet sich dabei immer 100 cm vom Fahrschlauch entfernt

und der Fußgänger ist von der Vorderkante des Verdeckungsfahrzeuges 70 cm entfernt. Um die Messfahrten so gut wie möglich zu reproduzieren, wurde der Fahrschlauch (208 cm) links und rechts mit Pylonen abgesteckt. Der Mittelpunkt des Ego-Fahrzeuges ist somit 204 cm vom Verdeckungsfahrzeug entfernt.

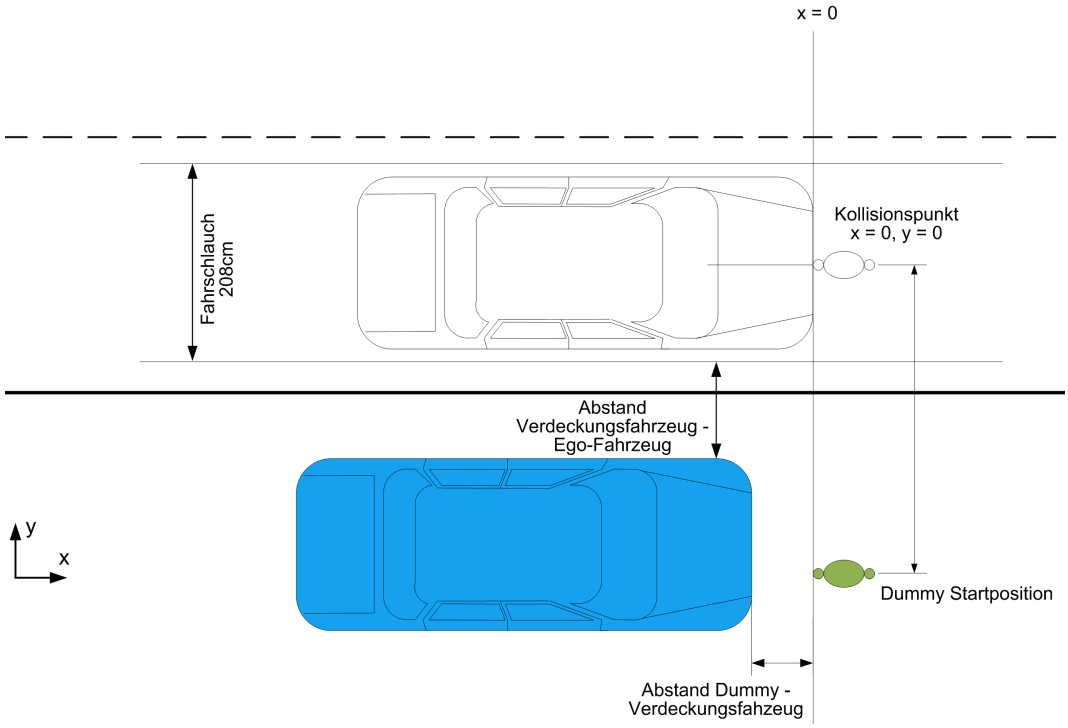

Abbildung 5.1: Versuchsaufbau am Fußgängerprüfstand

Da bei einzelnen Messungen z.B. in der Fahrzeuggeschwindigkeit, beim Kollisionspunkt oder bei der Fußgängererkennung Schwankungen auftreten können, werden alle Szenarien mindestens vier Mal durchgeführt und aus den einzelnen Messungen wird bei allen gemessenen Parametern der Mittelwert gebildet. Um die einzelnen Messungen zu vergleichen, sind die Messungen auf den Kollisionspunkt t = 0 normiert. Bei der Auswertung der Daten wurde der Zeitpunkt gesucht, bei dem die TTC gleich 2 Sekunden beträgt und auf -2 Sekunden gesetzt. Wenn das Fahrzeug nicht bremst, würde zu diesem Zeitpunkt die Kollision mit dem Fußgänger stattfinden.

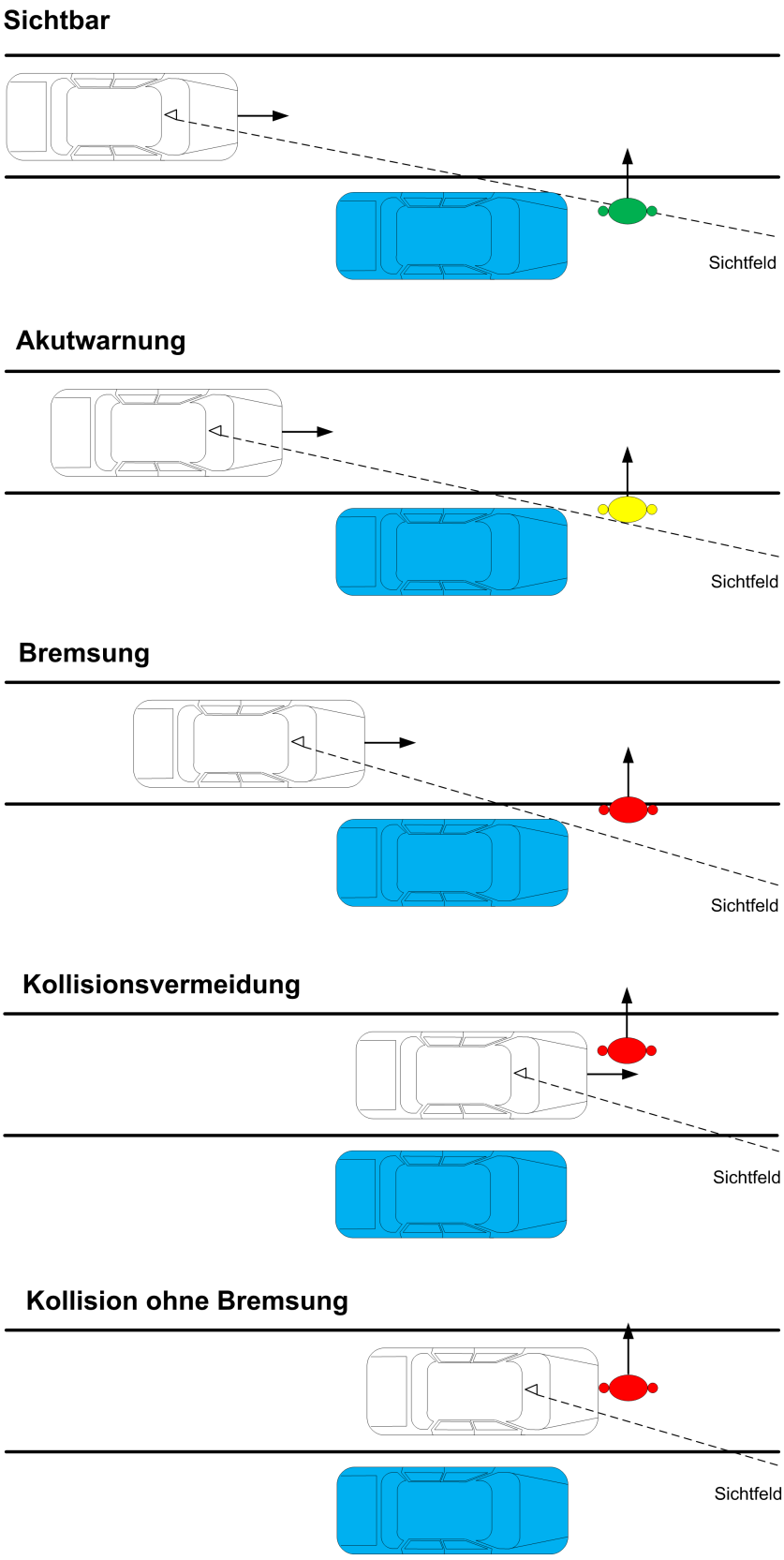

Abbildung 5.2: Schematischer Ablauf eines Szenarios

In Abbildung 5.2 ist der zeitliche Ablauf des Szenario 1 schematisch dargestellt, beginnend vom Zeitpunkt, ab wann der Fußgänger sichtbar ist, bis zum Stillstand des Fahrzeuges. Die dazugehörigen zeitlichen Verläufe des Abstandes in y-Richtung (oben) und x-Richtung (unten) des Fahrzeuges zum Fußgänger sind in Abbildung 5.3 dargestellt (schwarze Linie). Zusätzlich ist noch der Zeitpunkt, ab wann der Fußgänger sichtbar ist (grüne vertikale Linie), der Warnzeitpunkt (gelbe vertikale Linie) und der Bremszeitpunkt (rote vertikale Linie) in den beiden Diagrammen eingezeichnet. Beim schematischen Ablauf des Szenarios ist der Fußgänger je nach Situation (sichtbar/grün, Akutwarnung/gelb oder Bremsung/rot) gleich eingefärbt wie beim zeitlichen Verlauf. Die Diagramme in Abbildung 5.3 sind auf den Kollisionspunkt normiert (t = 0). Neben der hier dargestellten kamerabasierten Fußgängererkennung befinden sich in Kapitel 5.3.4 detailliertere Ergebnisse der Ko-TAG-basierten Fußgängererkennung und der Simulation.

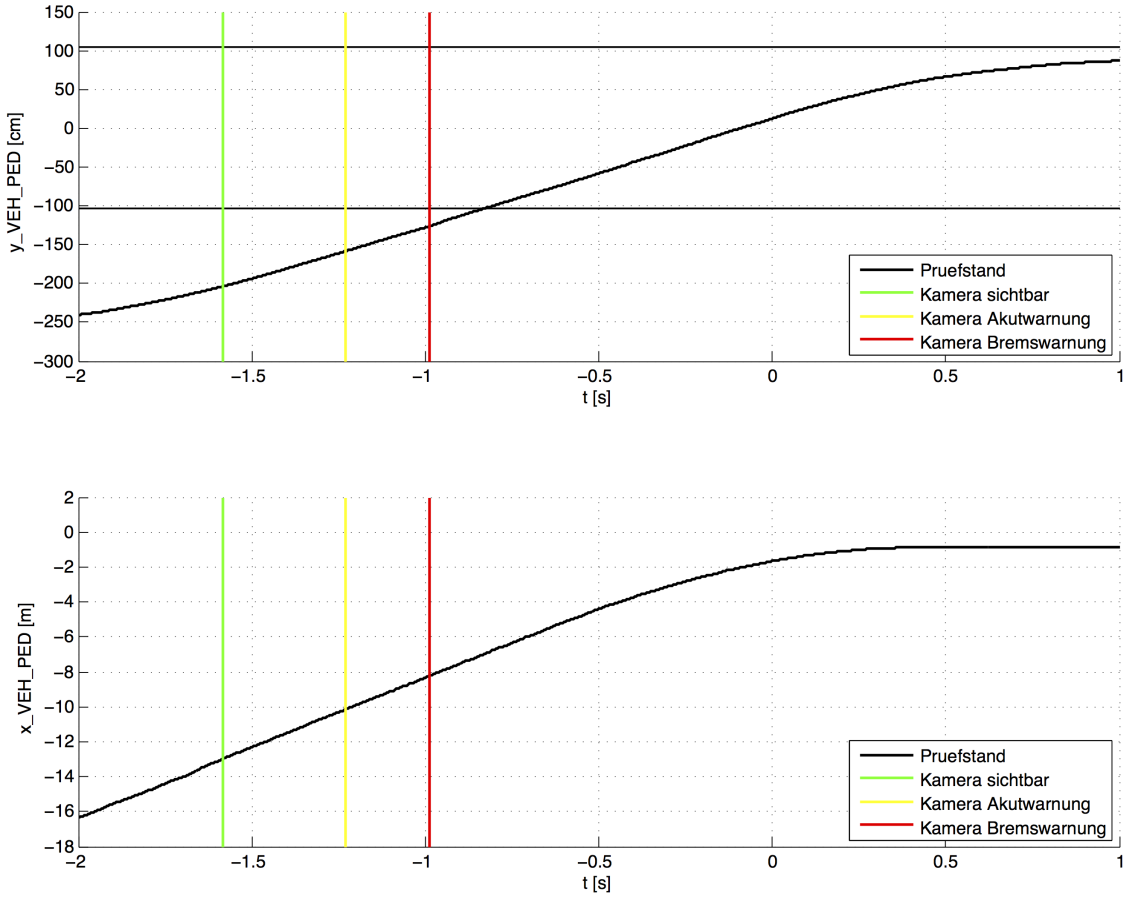

Abbildung 5.3: Zeitlicher Ablauf eines Szenarios

Das obere Diagramm stellt den y-Abstand des Fahrzeuges zum Fußgänger (y_VEH_PED) dar. Nachdem sich das Fahrzeug in y-Richtung nicht bewegt, entspricht der Verlauf dem

Weg des Fußgängers. Aufgrund der begrenzten Größe des Prüfstandes bewegt sich der Fußgänger nicht ständig linear in Richtung des Kollisionspunktes, sondern beschleunigt aus dem Stand und bewegt sich danach konstant mit der gewünschten Geschwindigkeit und bremst beim Erreichen des Kollisionspunktes wieder bis zum Stillstand ab. Daraus ergibt sich der nichtlineare Verlauf des Fußgängers, wie im Diagramm gut erkennbar ist.
Das untere Diagramm stellt den x-Abstand des Fahrzeuges bis zum Kollisionspunkt mit dem Fußgänger (x_VEH_PED) dar. Solange das Fahrzeug nicht bremst, nimmt der Abstand zum Fußgänger linear ab, bis die Bremsung eingeleitet wird, danach verringert sich die Geschwindigkeit stetig und die Dauer bis zur Kollision verlängert sich. Dabei bewegt sich der Fußgänger vom Kollisionspunkt weg und schlägt weiter links am Fahrzeug auf, als wenn das Fahrzeug nicht bremst, letzteres ist in Abbildung 5.2 bei *Kollision ohne Bremsung* dargestellt ist. Wenn keine Bremsung stattfindet, trifft das Fahrzeug den Fußgänger an der am Fußgängerprüfstand definierten Stelle.
In Abbildung A.1 befinden sich die Ergebnisse der restlichen vier Szenarien. Zusätzlich ist dort neben dem Zeitpunkt der Sichtbarkeit der Kamera (hellgrüne vertikale Linie) noch der Zeitpunkt der Sichtbarkeit des Ko-TAG-Sensors (dunkelgrüne vertikale Linie) eingezeichnet. Details zur Bestimmung der Sichtbarkeit werden im Kapitel 5.2.2 erläutert.
Aus den Messungen der fünf Szenarien geht hervor, dass bei Szenario 1, 4 und 5 der Kollisionspunkt nicht genau bei der Fahrzeugmitte liegt. Es kommt zu Abweichungen von bis zu 24 cm. Auch bei Szenario 2 kommt es zu einer Abweichung von 13 cm, die der Fußgänger zu weit in der Mitte aufschlagen würde. Bei Szenario 3 liegt der Kollisionspunkt exakt auf der definierten Stelle. Die Abweichung zu den definierten Kollisionspunkten bei den Szenarien wird in der Simulation durch die Startposition des Fußgängers berücksichtigt.

5.2 Datenauswertung der Messfahrten

5.2.1 Matlab Auswertung

Um die Auswertung von Messfahrten am Fußgängerprüfstand zu vereinfachen, wurden die drei unten angeführten Matlab-Skripte erstellt. Damit ist es möglich, die Fahrten automatisiert auszuwerten. Die aufgezeichneten Messdaten des Prüfstandes, der Kamera und des Ko-TAG-Sensors werden damit ausgewertet.

- *auswertungCAN(*can_path, y_road_side_veh, offset_start_ped, v_ped, v_ego*)*
  Mit Hilfe dieses Skripts werden die einzelnen Fahrten eines Szenarios ausgewertet. Hierfür werden die notwendigen Daten aus den CAN-Aufzeichnungen im Fahrzeug ausgelesen, fehlende Positionsdaten des Prüfstandes werden interpoliert, die Sichtbarkeit des Fußgängers für die Sensoren wird bestimmt und relevante Daten zur weiteren

Auswertung werden ermittelt. Als Ergebnis liefert das Skript die beiden Rückgabewerte **kafas** und **kotag**. In diesen beiden Variablen steckt die benötigte Information, um das gewählte Szenario als Gesamtes näher zu betrachten.
Um ein Szenario auszuwerten, ist es notwendig, den Pfad mit den gewünschten Messfahrten anzugeben. Die Initialisierungsparameter für die einzelnen Szenarien sind in folgender Liste angeführt:

```
Szenario 1: [kafas, kotag] = auswertungCAN('CAN_mitte_5kmh',2040,0,5,30)
Szenario 2: [kafas, kotag] = auswertungCAN('CAN_75_8kmh',2040,-470,8,30)
Szenario 3: [kafas, kotag] = auswertungCAN('CAN_25_8kmh',2040,470,8,30)
Szenario 4: [kafas, kotag] = auswertungCAN('CAN_ohne_verdeckung_mitte',0,0,5,30)
Szenario 5: [kafas, kotag] = auswertungCAN('CAN_ohne_verdeckung_mitte_3kmh',0,0,3,30)
```

- *analyseSzenarioKAFASKoTAG(*kafas, kotag*)*
  Um die Schwankungen, die bei den Messfahrten auftreten, zu minimieren, werden mehrere Messfahrten pro Szenario durchgeführt. Mit diesem Skript werden alle Messfahrten auf die gleiche Zeitbasis gebracht. Anschließend wird der Mittelwert über alle Fahrten berechnet und dabei die Werte für die Erkennung, Identifizierung, Akutwarnung und Bremswarnung bestimmt.
  Sowohl für die Kamera als auch für den Ko-TAG-Sensor wird der Fehler bei der x- und y-Entfernungsmessung in Abhängigkeit von der Distanz zum Fußgänger ermittelt, dafür wird aus allen Messfahrten der Mittelwert der Abweichungen gebildet.

- *analyseSzenarien()*
  Um den y-Verlauf des Fußgängers, den x-Verlauf des Fahrzeuges und die Sichtbarkeit des Fußgängers der einzelnen Szenarien gemeinsam darzustellen, wird dieses Skript verwendet. Die Daten dafür stammen aus der Auswertung der einzelnen Szenarien vom Skript **analyseSzenarioKAFASKoTAG()**. Mit dem Skript wird auch der x- und y-Fehler sowohl von der Kamera als auch vom Ko-TAG-Sensor über alle fünf Szenarien bestimmt. Hierfür wird der Fehler der einzelnen Szenarien verwendet und daraus der Mittelwert gebildet.

5.2.2 Auswertung und Berechnung der Sichtbarkeit

Die Sichtbarkeit des Fußgängers ist abhängig von der Positionierung des Sensors, je größer der Öffnungswinkel ist und je weiter vorne der Sensor platziert ist, desto früher ist der Fußgänger sichtbar. Zusätzlich wird der zeitliche Vorteil größer, je später der Fußgänger aus der Verdeckung kommt. Wie man in Abbildung A.1 erkennen kann, ist mit Ko-TAG der Fußgänger früher sichtbar. Abbildung 5.4 verdeutlicht den Unterschied durch die Platzierung der Kamera und des Ko-TAG-Sensors im Fahrzeug.

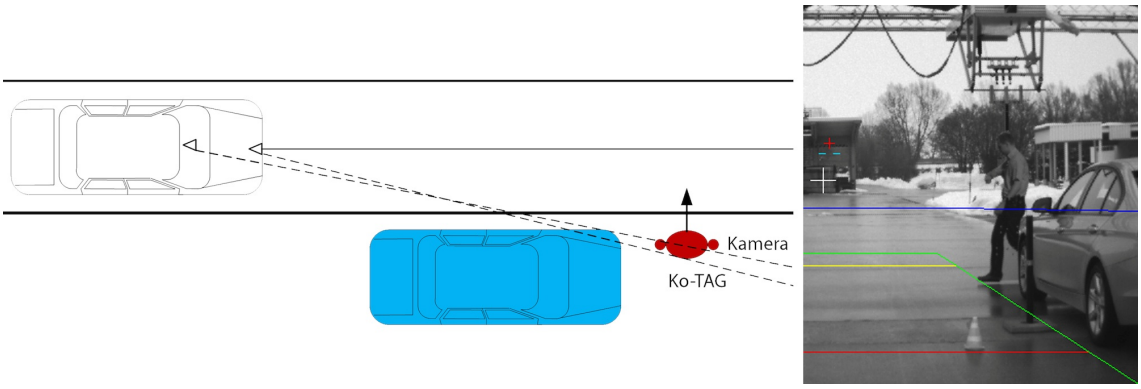

Abbildung 5.4: links: Unterschied der Sichtbarkeit zwischen Kamera und Ko-TAG
rechts: Zeitpunkt, ab wann der Fußgänger als sichtbar gilt

Um die aufgezeichneten Messdaten automatisiert auszuwerten, werden die oben angeführten Matlab-Skripte verwendet. Eine zu analysierende Größe ist dabei die Sichtbarkeit des Fußgängers. Dafür wird die genaue Position des Fußgängers zum Ego-Fahrzeug und die Position des Verdeckungsfahrzeuges benötigt.
Folgender Auszug aus dem Matlab Skript **auswertungCAN()** zeigt die Berechnung des Zeitpunktes, ab wann der Fußgänger für die Kamera und Ko-TAG sichtbar ist. Bei der Berechnung wird der Winkel zur Kante des Verdeckungsfahrzeuges und der Winkel zum Rücken des Fußgängers bestimmt und miteinander verglichen. Sobald der Winkel zum Fußgänger kleiner ist, gilt der Fußgänger als sichtbar.

```
for( i= 1:length(time_vector) )
    x_pos_veh = x_pfgs_int(i);    %Abstand Ego-Fahrzeug - Fußgänger in x-Richtung (vom Pruefstand)
    y_pos_veh = y_pfgs_int(i);    %Abstand Ego-Fahrzeug - Fußgänger in y-Richtung

    %Berechnung aktueller Winkel Kamera-Fußgänger, Kamera-Verdeckungsfahrzeug
    angle_camera_veh_roadside = atan(Y_POSITION_VEH_ROADSIDE/(-x_pos_veh+X_POSITION_CAMERA
                        +X_POSITION_VEH_ROADSIDE))*57.2;
    angle_camera_ped = atan(abs(y_pos_veh-WIDTH_PED/2)/(-x_pos_veh+X_POSITION_CAMERA))*57.2;

    diff_angle =  angle_camera_veh_roadside - angle_camera_ped;
    if( angle_camera_ped < angle_camera_veh_roadside && abs(angle_camera_ped) < 19
                        && t_visible_camera == 0 )
        t_visible_camera = time_vector(i);
    end

    %Berechnung aktueller Winkel Ko-TAG-Fußgänger, Ko-TAG-Verdeckungsfahrzeug
    % ...

end
```

Die Messungen am Prüfstand haben gezeigt, dass der Fußgänger für die Kamera sichtbar ist, wenn der Mittelpunkt des Fußgängers 10 cm vom Öffnungswinkel nach innen ent-

fernt ist. Obwohl der Oberkörper des Fußgängers eine größere Breite hat, ist aufgrund der Verdeckung, wie in Abbildung 5.4 (rechts) dargestellt ist, der Fußgänger bis auf den Unterschenkel des hinteren Beines komplett sichtbar. Aus diesem Grund wurde die Fußgängerbreite mit 20 cm festgelegt.

5.3 Mess- und Simulationsergebnisse

In diesem Abschnitt sind die Messergebnisse der fünf Szenarien mit der kamerabasierten und Ko-TAG-basierten Fußgängererkennung angeführt. Für das Szenario 1 sind die Mess- und Simulationsergebnisse detailliert grafisch dargestellt und genauer beschrieben, die Interpretation der anderen Messergebnisse erfolgt nach dem selben Prinzip. Für die restlichen Szenarien befinden sich die Ergebnisse im Anhang unter A Mess- und Simulationsergebnisse.
Bei den Versuchsfahrten wurden die Daten der Kamera und von Ko-TAG aufgezeichnet. Es war dabei aber nur möglich, den Bremseingriff durch den Ko-TAG-basierten Fußgängerschutz zu bestimmen, da die Bremsschnittstelle entweder von der Kamera oder von Ko-TAG angesteuert werden kann, beide Systeme können bei einer Fahrt nicht zugleich aktiviert sein. Deshalb wurden alle Messungen mit dem Bremseingriff durch Ko-TAG durchgeführt. Dies erfolgte einerseits, um den Funktionsalgorithmus zu testen und andererseits um alle Messungen besser miteinander vergleichen zu können. Dadurch konnte aber nicht der tatsächliche Bremszeitpunkt der kamerabasierten Fußgängererkennung aufgezeichnet werden. Um die Simulation der kamerabasierten Fußgängererkennung mit den Messergebnissen zu vergleichen, wurde der Bremszeitpunkt von Ko-TAG aus den Messungen ausgewertet und für die Simulation verwendet.

5.3.1 Parametrisierung Bremsverlauf

Die Parametrisierung des Bremsmodelles erfolgt mit Hilfe der Messungen am Fußgängerprüfstand. Dafür werden die fünf Messungen des Geschwindigkeitsverlaufes (v_VEH) aus Szenario 1 verwendet, die in Abbildung 5.5 dargestellt sind. Die dünnen Linien stellen die einzelnen Messungen dar, die vertikalen Linien zeigen die dazugehörigen Bremsanforderungen. Der gemittelte Geschwindigkeitsverlauf ist die dickere blaue Linie, die linke vertikale blaue Linie ist der gemittelte Zeitpunkt der Bremsanforderung und die rechte vertikale blaue Linie stellt den gemittelten Stillstand des Fahrzeuges dar. Die durchgeführten Messungen sind dabei auf den Zeitpunkt TTC = 2 s normiert.

Aus den Messungen lässt sich erkennen, dass die zeitliche Schwankung der Bremsanforderung im gleichen Bereich schwankt wie der Fahrzeugstillstand, der Verlauf der einzelnen

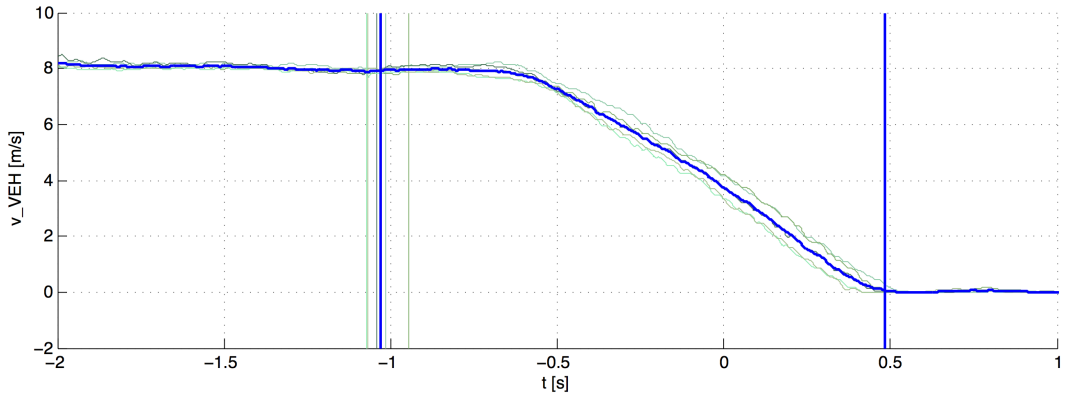

Abbildung 5.5: Geschwindigkeitsverlauf des Ego-Fahrzeuges aller Messfahrten - Szenario 1

Messungen ist zueinander identisch. Die Schwankung der Bremsanforderung ist auf den Funktionsalgorithmus zurückzuführen. Bei jeder Messung variiert die Anfahrtsgeschwindigkeit des Ego-Fahrzeuges und die Position des Fußgängers. Zusätzlich liefert der Funktionsalgorithmus nicht kontinuierlich Werte, sondern nur alle 50 ms erfolgt eine Neuberechnung der Situation und dadurch kann eine Bremsansteuerung nur in dieser Update-Rate angefordert werden. Deshalb wird aus allen fünf Messungen von Szenario 1 der Mittelwert bestimmt und daraus das Modell parametrisiert.
Beim Fahrzeug kommt es zu keiner Vorkonditionierung der Bremse, somit kann der Bremsverlauf näherungsweise einfach modelliert werden. Aus der Auswertung des gemittelten Bremsvorganges aller Einzelmessungen ergeben sich für die Simulation folgende Parameter:

```
bremse_totzeit: 0.22 s   % Dauer bis die Verzögerung aufgebaut wird
bremse_a_max = -8.1 m/s^2   % maximale Verzögerung
bremse_delta_rampe: 20 % a_max / delta_rampe = Dauer bis die max. Verzögerung aufgebaut wird
```

Der Vergleich des Geschwindigkeitsverlaufes in Szenario 1 zwischen der Simulation und der gemittelten Messung ist in Abbildung 5.6 dargestellt. Die blaue Linie zeigt den Geschwindigkeitsverlauf der Messung und die rote den Geschwindigkeitsverlauf der Simulation. Die rote vertikale Linie ist der Zeitpunkt der Bremsanforderung und die blaue vertikale Linie ist der Stillstand des Fahrzeuges.

In Abbildung 5.7 ist der x-Abstand des Fahrzeuges zum Fußgänger dargestellt, der blaue Verlauf ist die Messung durch den Prüfstand und der rote Verlauf die Simulation, die vertikale rote Linie zeigt den Zeitpunkt der Bremsanforderung und die schwarze vertikale Linie den Zeitpunkt des Fahrzeugstillstandes. Zwischen der Simulation und der Messung aus Szenario 1 gibt es eine Abweichung von 2 cm, um die das Fahrzeug bei der Simulation später zu stehen kommt.

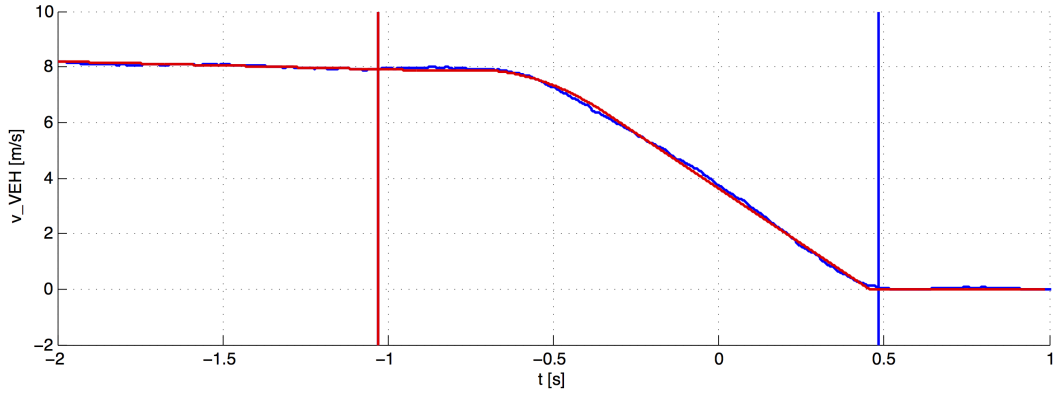

Abbildung 5.6: Vergleich des Geschwindigkeitsverlaufes - Messung (blau) und Simulation (rot) - Szenario 1

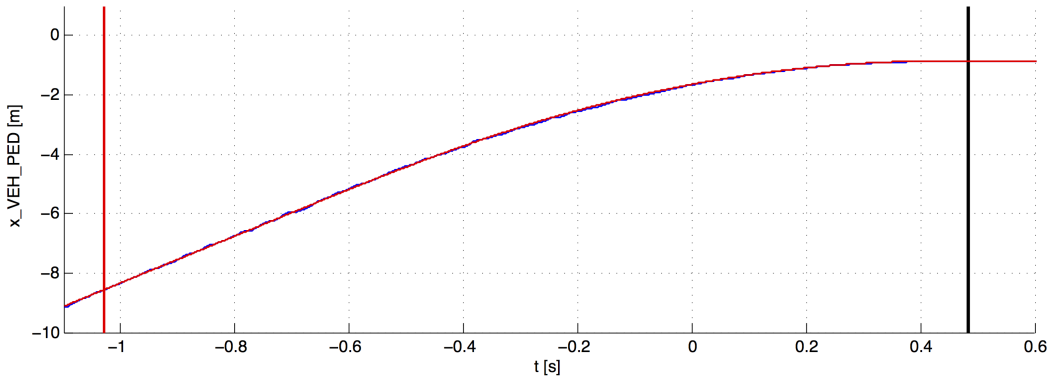

Abbildung 5.7: Vergleich x-Verlauf zwischen Messung (blau) und Simulation (rot) - Szenario 1

Während der Messungen war die Straße nass, somit konnte das Fahrzeug nur mit einer maximalen Verzögerung von 8.1 m/s^2 bremsen. Beim Bremsmodell ändert sich je nach Untergrund der Wert der maximalen Verzögerung a_max, bei trockener Straße wäre der Bremswert höher.
Mit den ermittelten Simulationsparametern aus Szenario 1 wurde der Bremsverlauf von Szenario 4 simuliert. Das Ergebnis des x-Verlaufes des Fahrzeuges ist in Abbildung 5.8 dargestellt, dabei kommt das Fahrzeug bei der Simulation um 4 cm später zu stehen.

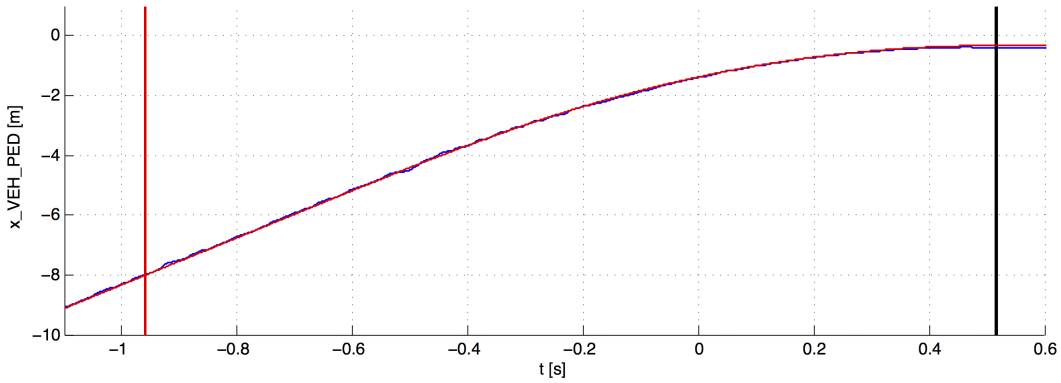

Abbildung 5.8: Vergleich x-Verlauf zwischen - Messung (blau) und Simulation (rot) - Szenario 4

5.3.2 Auswertung Kamerafehler

Der Fehler der Kamera bei der Positionsschätzung in x- und y-Richtung wird durch den Mittelwert aller gemessenen Fahrten am Prüfstand bestimmt. Mit Hilfe des Matlab-Skripts **analyseSzenarien()** werden die Variablen **camera_error_x** und **camera_error_y** gebildet, die direkt in der Simulation im Kameramodell verwendet werden. In Abbildung 5.9 ist der Messfehler der Kamera, resultierend aus der tatsächlichen Position des Fußgängers und der gemessenen Position durch die Kamera dargestellt. Im oberen Diagramm ist der y-Fehler und im unteren Diagramm der x-Fehler jeweils in Abhängigkeit zur Entfernung zwischen Kamera und Fußgänger abgebildet. In den Diagrammen kann man gut erkennen, dass sich bei den einzelnen Szenarien der x- und y-Fehler in die gleiche Richtung bewegt, nur der Betrag unterscheidet sich dabei.
Die Ergebnisse aus Szenario 3 sind nicht in die Bestimmung der Fehler mit eingeflossen, da bei den Einzelmessung keine brauchbaren Messergebnisse geliefert wurden. Aufgrund starker Schwankung bei der Erkennung des Fußgängers kommt es nur zu einem sehr kurzen Fenster, in dem alle vier Messungen Werte liefern.
Betrachtet man den Mittelwert des y-Fehlers, so ist dieser wie erwartet kleiner als der x-Fehler. Mit der Kamera lässt sich der y-Abstand systembedingt genauer bestimmen.
Der maximale y-Fehler im gemessenen Bereich beträgt weniger als 20 cm bei allen Messungen. Bei der x-Messung kommt es zu wesentlich stärkeren Schwankungen bei den einzelnen Messungen, Abweichungen von bis zu 280 cm treten auf, jedoch liegt der Fehler im relevanten Bereich (TTC $<$ 1.5 s) unter 100 cm. Im Bereich zwischen -5 m und -1 m kommt es wieder zu einem Anstieg des Fehlers, jedoch spielt in diesem Bereich der Fehler keine Rolle.

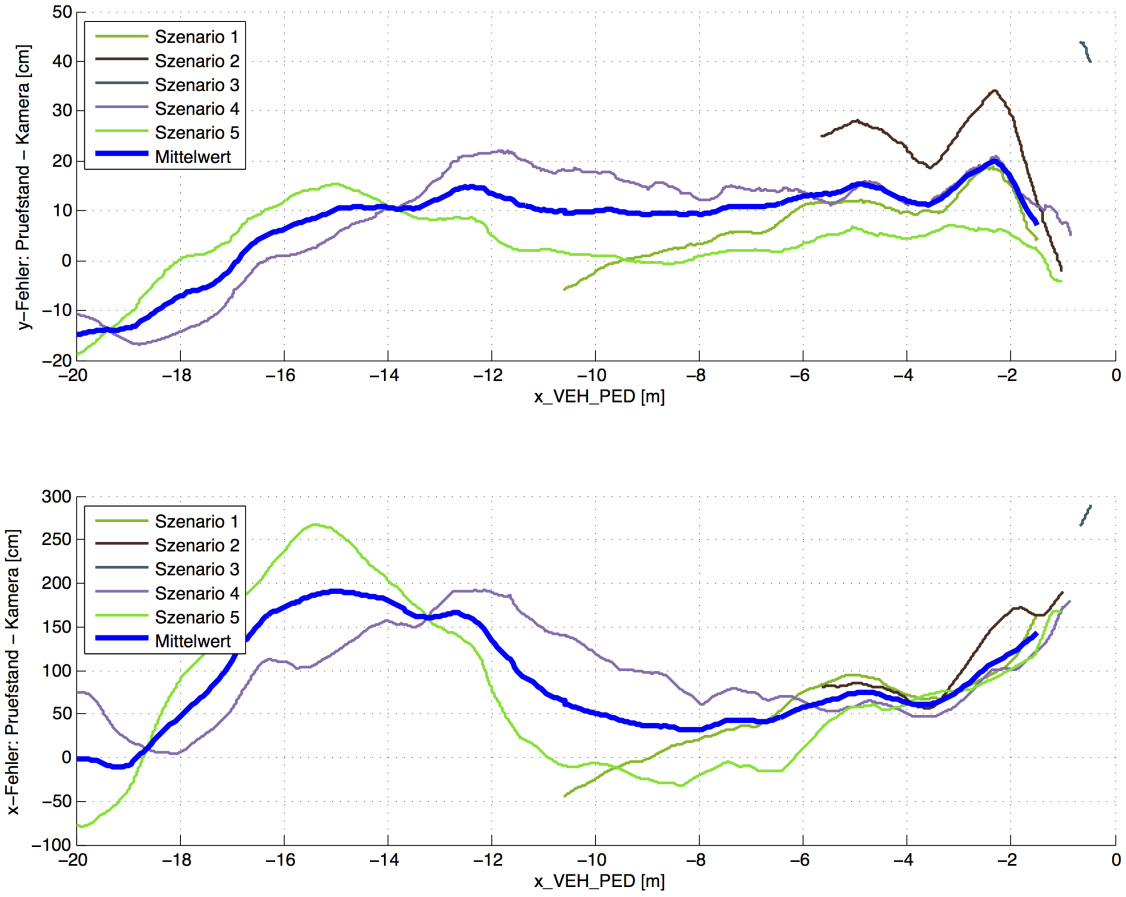

Abbildung 5.9: y/x-Fehler der Kamera, gemittelt über alle Messungen

5.3.3 Auswertung Ko-TAG-Fehler

Die Auswertung des Fehlers beim Ko-TAG-Sensor erfolgt nach dem selben Prinzip wie bei der Kamera mit dem Matlab-Skript **analyseSzenarien()**. Die Messfehler des Ko-TAG-Sensors in y- und x-Richtung in Abhängigkeit zur Entfernung zwischen dem Sensor und dem Fußgänger sind in Abbildung 5.10 dargestellt. Im oberen Diagramm ist der y-Fehler und im unteren Diagramm der x-Fehler abgebildet.

Bei der Entfernungsmessung in x-Richtung liefert der Sensor ab einer Entfernung von 14 m zuverlässige Werte, davor weisen die Szenarien starke Schwankungen auf. Der Anstieg des Fehlers kurz vor Aufprall ist auf das Hochklappen des Fußgängers zurückzuführen. Damit es zu keiner Kollision zwischen Fahrzeug und Fußgänger kommt, klappt dieser automatisch 0.4 s vor einer bevorstehenden Kollision hoch. Die laterale Entfernungsmessung der einzelnen Szenarien schwankt deutlich stärker, es kommt bei Entfernungen um 18 m zwischen Sensor und Fußgänger zu Abweichungen von ±1.5 m. Erst unter einer Entfernung von 16 m werden die Messungen genauer. Vergleicht man die fünf gemessenen Szenarien,

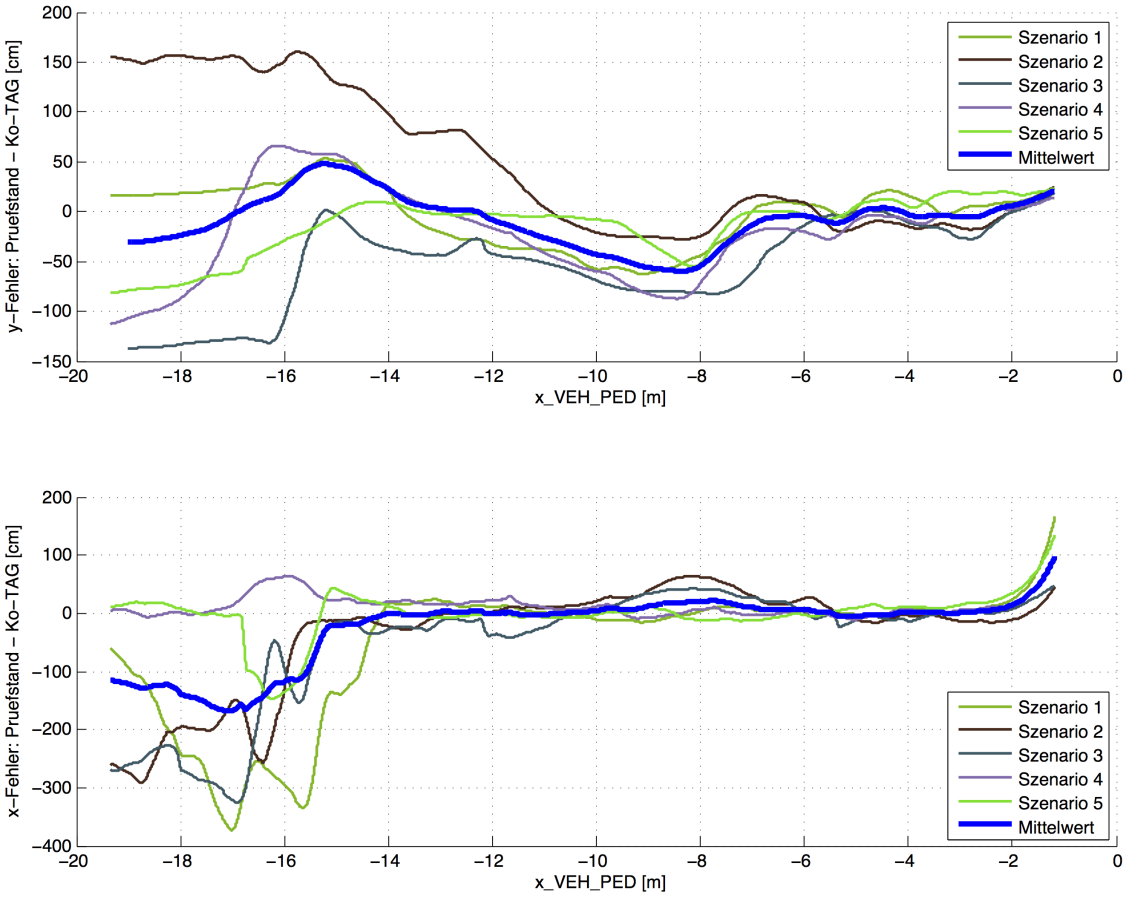

Abbildung 5.10: y/x-Fehler des Ko-TAG-Sensors, gemittelt über alle Messungen

so kann man einen ähnlichen Trend bei der Entfernungsmessung in y-Richtung erkennen. Betrachtet man die einzelnen Verläufe des Fehlers bei der Entfernungsmessung in x- und y-Richtung in Abbildung 5.10, so kann kein Zusammenhang zwischen verdeckt und sichtbar herausgelesen werden, deshalb wurde wie ursprünglich im Ko-TAG-Modell (Kapitel 3.4 Modellierung Ko-TAG) nicht berücksichtigt, ob der Fußgänger für den Sensor sichtbar oder verdeckt ist. Bei der Simulation wird nur der gemittelte Fehleraufschlag in Abhängigkeit zur Entfernung verwendet und nicht berücksichtigt, wie weit der Fußgänger sich hinter der Fahrzeugkante befindet.

5.3.4 Szenario 1

Durch die Auswertung der Bewegungsverläufe des Fahrzeuges und des Fußgängers, die in der Abbildung A.1 dargestellt ist, wurden die Startparameter des Fußgängers und des Fahrzeuges für die Simulation bestimmt. Das Szenario wird durch die in folgender Liste angeführten Parameter vollständig definiert.

Simulationsparameter:

```
a_veh_start: -0.29 m/s^2   %Fahrzeug bewegt sich nicht mit konstanter Geschwindigkeit
v_veh_start: 29.5 km/h (t = -2 s) % Geschwindigkeit des Fahrzeuges 2 s vor Kollision
v_ped: 4.93 km/h % Fußgängergeschwindigkeit
y_ped_start: 2.64 m % Startposition des Fußgängers vom Fahrbahnmittelpunkt
x_veh_start: -18.83 m % Startposition des Fahrzeug-Nullpunktes vom Kollisionspunkt
x_veh_roadside: -3.33 m % x-Abstand des Verdeckungsfahrzeuges
y_veh_roadside: 2.97 m % y-Abstand des Verdeckungsfahrzeuges

Kamera:
Update-Rate Funktionsalgorithmus: 0.094 s
Bremszeitpunkt in der Simulation:  1.06 s   % Auslösung durch Ko-TAG bei der Messfahrt
Ko-TAG:
Update-Rate Funktionsalgorithmus: 0.05 s
```

Kamerabasierte Fußgängererkennung:

Die Ergebnisse der Messungen am Prüfstand und der Simulation des Szenario 1 befinden sich in Tabelle 5.2, für die grafische Darstellung wurde der Mittelwert der durchgeführten Messungen gebildet und als Vergleich für die Simulation herangezogen. In der Tabelle stehen die fünf Messungen vom Prüfstand, gefolgt vom daraus resultierenden Mittelwert der einzelnen Messergebnisse, den Werten aus der Simulation und der Differenz zwischen Messung und Simulation. In der ersten Spalte der Tabelle steht die Bezeichnung der Messung, danach ist die Tabelle in vier Blöcke unterteilt: sichtbar, erkannt, Akutwarnung und Bremsung. Dazu ist jeweils der Zeitpunkt, wann das Ereignis eingetreten ist und der dazugehörige Abstand in y- und x-Richtung zwischen Fahrzeug und Fußgänger in der Tabelle eingetragen.

Nr.	sichtbar [s]	y_PED [m]	x_VEH [m]	erkannt y_PED [m]	erkannt x_VEH [m]	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
103846	-1.53	-2.04	-12.61	-1.82	-11.36	-1.28	-1.70	-10.65	-0.87	-1.14	-7.42
104406	-1.63	-2.03	-13.71	-1.68	-10.76	-1.32	-1.69	-10.77	-1.03	-1.30	-8.45
104551	-1.57	-2.03	-13.43	-1.69	-11.14	-1.09	-1.40	-9.42	-0.88	-1.08	-7.63
104833	-1.56	-2.04	-12.64	-1.69	-10.95	-1.16	-1.51	-9.38	-1.06	-1.37	-8.57
105117	-1.61	-2.04	-12.98	-1.77	-11.95	-1.28	-1.59	-10.40	-1.09	-1.34	-8.93
MW	-1.58	-2.04	-13.07	-1.76	-11.23	-1.23	-1.58	-10.12	-0.99	-1.26	-8.24
Sim.	-1.57	-2.03	-12.94	-1.75	-11.15	-1.25	-1.60	-10.36	-0.97	-1.21	-8.10
Diff.	0.01	0.01	0.11	0.01	0.08	-0.02	-0.02	-0.24	0.02	0.05	0.14

Tabelle 5.2: Mess- und Simulationsergebnisse der Kamera

Der grafische Verlauf des Fußgängers und des Fahrzeuges ist in Abbildung 5.11 dargestellt. Im oberen Diagramm ist der y-Abstand zwischen Fahrzeug und Fußgänger, gemessen durch den Prüfstand (schwarze Linie) und der Abstand aus der Simulation (rote Linie) abgebildet. Zusätzlich zeigt das Diagramm die geschätzte Position des Fußgängers ermittelt durch

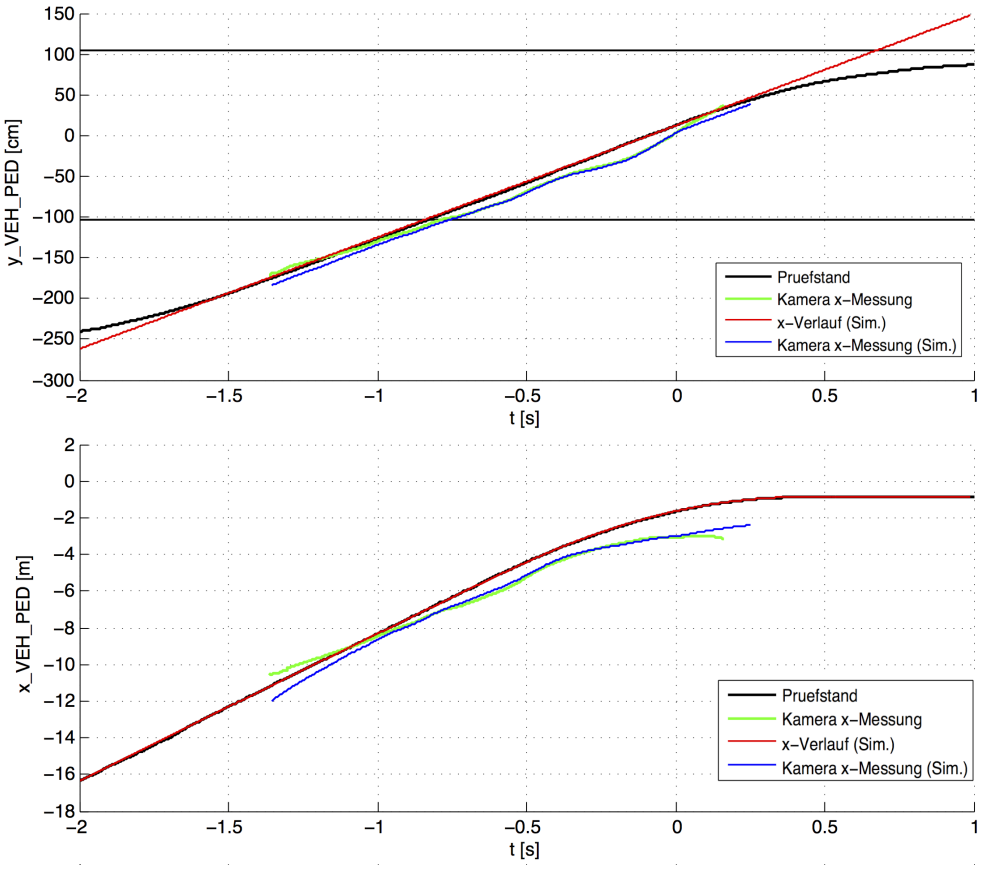

Abbildung 5.11: y/x-Verlauf des Fußgängers mit der Kamera, Messung und Simulation

die Kamera (grüne Linie) und aus der Simulation (blaue Linie). Die Sichtbarkeit bei der Messung (grüne Linie), die Akutwarnung (gelbe Linie), die Bremswarnung (rote Linie) aus der Messung am Prüfstand und die dazugehörigen Ergebnisse aus der Simulation (strichlierte Linien) sind in Abbildung 5.12 dargestellt.

In Abbildung 5.13 ist die gemessene Abweichung der geschätzten Positionen des Fußgängers von der Kamera zur tatsächlichen Position durch die Messung am Fußgängerprüfstand dargestellt. Die Abweichung liegt bei der y-Messung bei $\pm$ 5 cm. Eine größere Abweichung tritt erst auf, sobald das Fahrzeug stärker verzögert (-0.5 s). Die Kamera scheint die Reduktion der Eigengeschwindigkeit nicht zu berücksichtigen, jedoch spielt die Abweichung in diesem Bereich keine Rolle, da die Warnung und Bremsung eingeleitet wurde.

Vergleicht man die Simulation mit der realen Messung, so kommt es wie erwartet zu Abweichungen. Die erste Abweichung tritt bei der Simulation des Zeitpunktes der Sichtbarkeit auf, hier liegt das Problem vor, dass sich der Fußgänger am Prüfstand nicht konstant be-

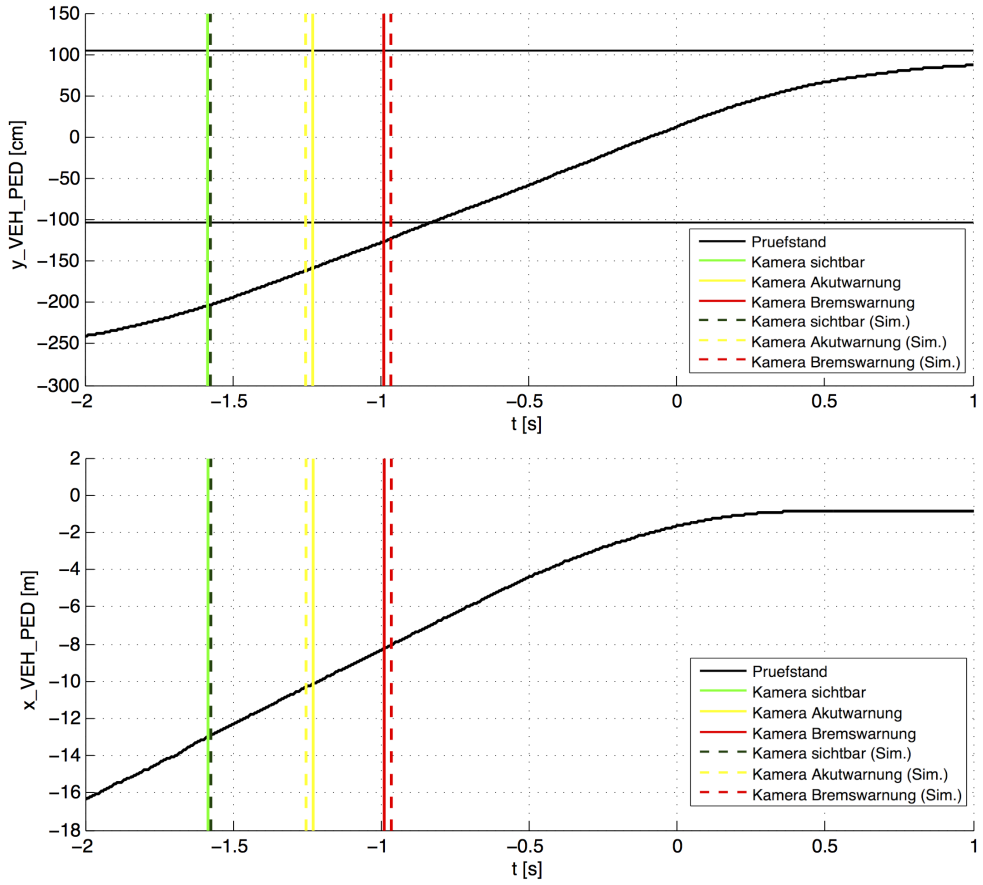

Abbildung 5.12: Sichtbarkeit und Warnzeitpunkte mit der Kamera, Messung und Simulation

wegt. In der Simulation wurde deshalb der Verlauf des Fußgängers so gewählt, dass die Abweichung der Position des Fußgängers zwischen Prüfstand und Simulation im kritischen Bereich (-1.5 s bis -1 s vor der Kollision) am geringsten ist.
Auch bei den Zeitpunkten der Akut- und Bremswarnung kommt es zu Abweichungen bei der Simulation. Einerseits ist dies auf die Modellierung der Entfernungsschätzung durch das Kameramodell zurückzuführen und andererseits auf den Zeitpunkt, zu dem die Kamera Daten für den Funktionsalgorithmus liefert. Abhängig von der aktuellen Fahrzeugposition und dem Zeitpunkt, zu dem die Kamera Daten liefert, kommt es zu einer maximalen zeitlichen Abweichungen von der halben Update-Rate der Kamera. Bei der Simulation liegt die Abweichung der beiden Warnpunkte zur Messung bei maximal 20 ms.

Ko-TAG-basierte Fußgängererkennung:
Verglichen mit den Messwerten der Kamera ist für Ko-TAG der Fußgänger auch sichtbar, wenn dieser noch verdeckt ist. In der grafischen Darstellung in Abbildung 5.14 ist dies

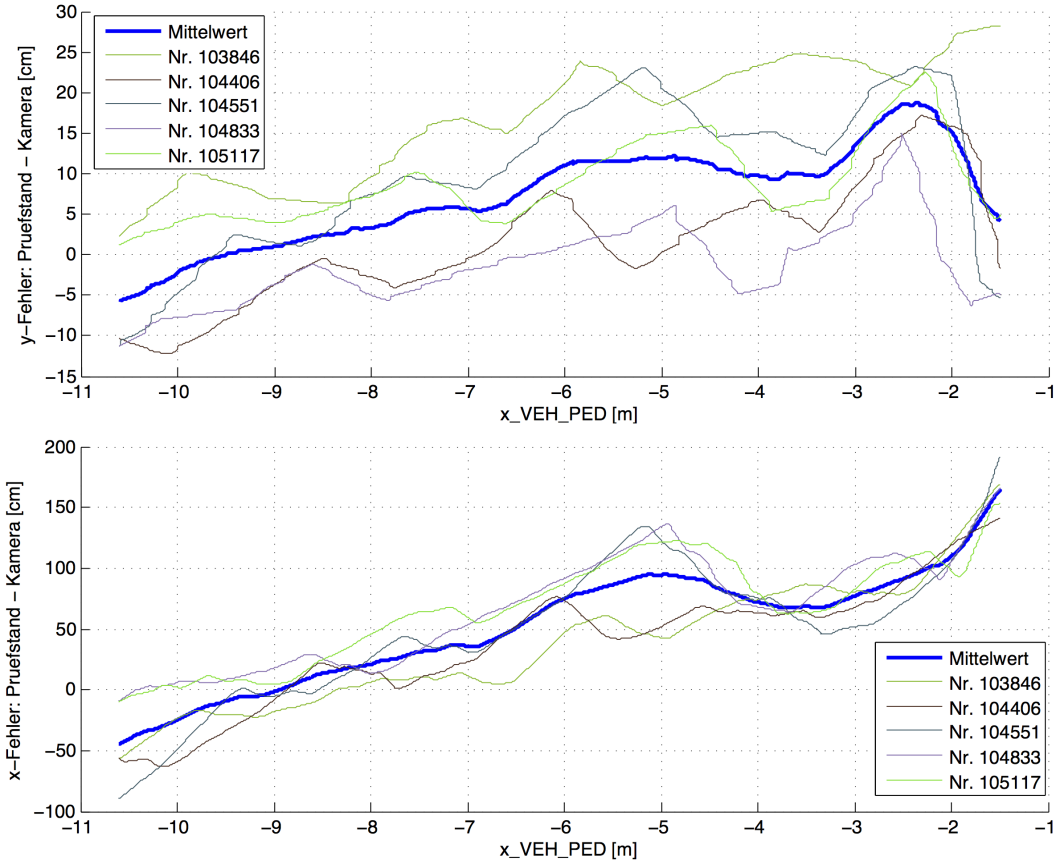

Abbildung 5.13: y/x-Verlauf der Fehlerabweichung der Kamera

gut ersichtlich, der Sensor liefert immer Positionsdaten des Fußgängers. Die dazugehörigen Messwerte des Szenarios befinden sich in Tabelle 5.3, der Aufbau der Tabelle ist der selbe wie bei der kamerabasierten Fußgängererkennung im vorigen Abschnitt. Bei der Abbildung des Szenarios sind neben den Positionsverläufen zwischen Fahrzeug und Fußgänger in y- und x-Richtung auch die Sichtbarkeit und die Warn-/Bremszeitpunkte der Messung und der Simulation abgebildet. Die Beschriftung der Diagramme erfolgt nach dem selben Prinzip wie bei der kamerabasierten Fußgängererkennung im vorigen Abschnitt. Zusätzlich wird im Diagramm noch der Verlauf der TTC (blaue Linie) dargestellt. Im Gegensatz zur Kamera liefert Ko-TAG eine genauere Entfernungsmessung in x-Richtung, dafür kommt es bei der Entfernungsmessung zu einer stärkeren Abweichung in y-Richtung. Solange der Fußgänger nicht sichtbar ist, wird dieser weiter rechts vom Fahrbahnrand gemessen. Kurz bevor sich der Fußgänger aus der Verdeckung bewegt, ändert sich dieses Verhalten und der Fußgänger wird vom Sensor näher im Fahrschlauch bestimmt. Bei einer Distanz von unter 8 m pendelt sich ein Fehler von kleiner 30 cm ein.

Nr.	sichtbar [s]	y_PED [m]	x_VEH [m]	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
103846	-1.55	-2.06	-12.82	-1.43	-1.90	-11.90	-1.01	-1.39	-8.56
104406	-1.65	-2.05	-13.48	-1.41	-1.78	-11.50	-1.05	-1.33	-8.74
104551	-1.60	-2.05	-13.52	-1.45	-1.90	-12.32	-1.04	-1.32	-9.04
104833	-1.58	-2.06	-12.74	-1.49	-1.96	-12.07	-1.07	-1.38	-8.65
105117	-1.63	-2.05	-13.26	-1.25	-1.55	-10.17	-1.06	-1.30	-8.76
MW	-1.60	-2.05	-13.16	-1.40	-1.82	-11.59	-1.05	-1.32	-8.75
Sim.	-1.58	-2.06	-13.03	-1.41	-1.81	-11.61	-1.0	-1.26	-8.36
Diff.	0.02	-0.01	0.13	-0.01	0.01	-0.02	0.05	0.06	0.39

Tabelle 5.3: Mess- und Simulationsergebnisse des Ko-TAG-Sensors

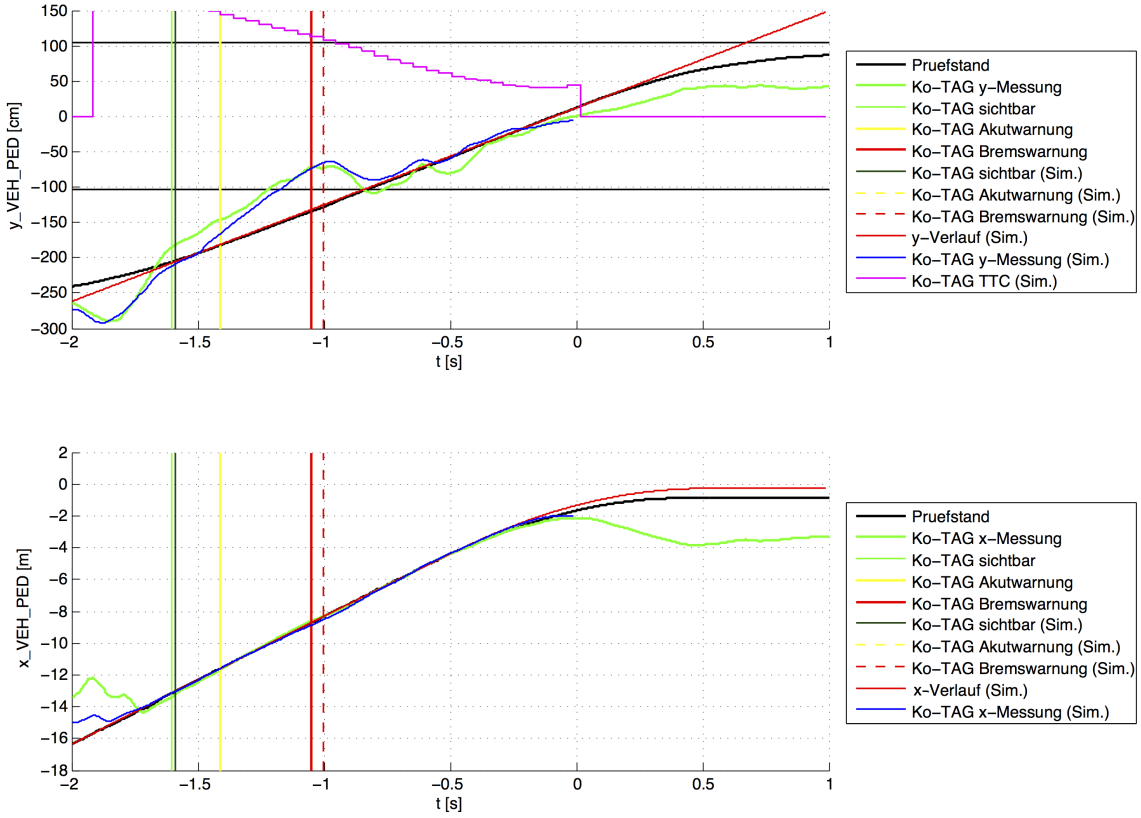

Abbildung 5.14: y/x-Verlauf des Fußgängers mit Ko-TAG, Messung und Simulation - Szenario 1

Bei der Simulation der Sichtbarkeit kommt es aufgrund des oben beschriebenen Bewegungsverlaufes des Fußgängers zum gleichen Fehler wie bei der kamerabasierten Simulation. Zwischen Simulation und Messung tritt bei der Akutwarnung nur eine Abweichung von 10 ms auf. Die Akutwarnung wird bei der Simulation wie erwartet ausgelöst, wenn die TTC kleiner 1.5 s ist und der seitliche Abstand des Fußgängers zur Fahrzeugmitte von 174 cm unterschritten wird.

Betrachtet man die Messfahrten am Prüfstand, so müsste die Akutwarnung bereits früher erfolgen, da der Fußgänger den definierten Abstand früher unterschritten hat. Der Zeitpunkt der Akutwarnung ergibt sich aber aus den fünf gemittelten Messungen und dabei streuen sowohl der Warnzeitpunkt als auch die Abstände zum Fußgänger.

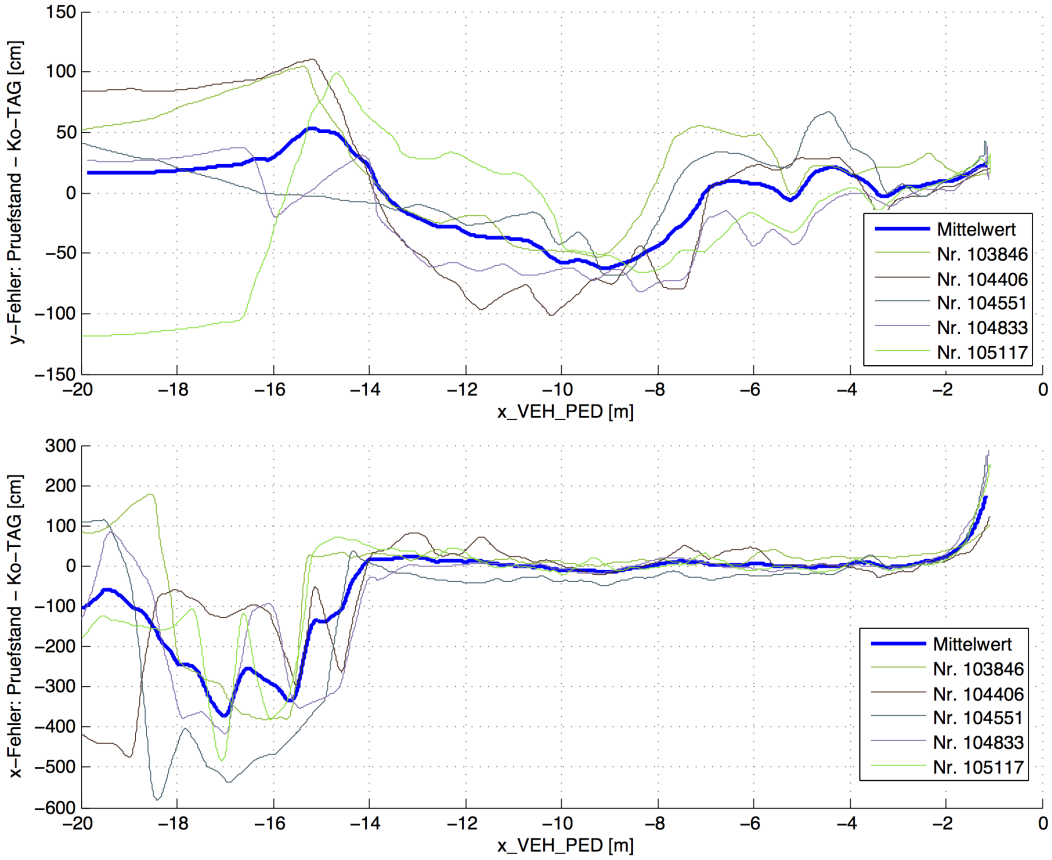

Abbildung 5.15: y/x-Verlauf der Fehlerabweichung bei Ko-TAG

Beim Bremseingriff kommt es zu einer Abweichung von 50 ms, dies ist auf die Abweichung bei der Entfernungsmessung in x-Richtung zurückzuführen. In Abbildung 5.14 ist der Grund dafür gut ersichtlich: Bei einer Distanz von 9 m zwischen Fahrzeug und Fußgänger wird die Entfernung zum Fußgänger in der Simulation um 25 cm größer bestimmt. Dadurch kommt es zu unterschiedlichen TTC Berechnungen und somit wird erst im nächsten Zeitschlitz eine Bremsung vom Algorithmus angefordert.
Die Messung in y-Richtung spielt zu diesem Zeitpunkt keine Rolle, da sich der Fußgänger bereits im Fahrschlauch befindet und eine Ungenauigkeit bei der Entfernungsmessung keine Auswirkung hat.

5.3.5 Szenario 2 - 5

Die Ergebnisse der Messungen am Prüfstand und der Simulation von Szenario 2 bis 5 befinden sich im Anhang A.

5.3.6 Identifikationszeit des Fußgängers durch die Kamera

Im Gegensatz zu Ko-Tag kann der Fußgänger durch die Kamera erst erkannt werden, wenn er sich im Sichtfeld der Kamera befindet. Die Kamera ist nicht in der Lage, den Fußgänger sofort zu identifizieren und die Gefahrensituation einzuschätzen, erst nach mehreren Frames kann eine genaue Situationsbestimmung erfolgen.
Bei den Messungen war Ko-TAG bei jeder Messfahrt mit dem Transponder am Fußgänger verbunden, ein Datenaustausch zwischen beiden war möglich und somit konnte die Position bestimmt werden. Besonders wenn der Fußgänger mit erhöhter Geschwindigkeit (8 km/h) in den Fahrschlauch läuft, hat die Kamera den Nachteil, dass sich der Fußgänger bereits innerhalb des Warnkorridors befindet, aber noch nicht erkannt wird und somit die Warnung erst zu einem späteren Zeitpunkt ausgegeben wird. Besonders bei Szenario 3, wo der Fußgänger bei 25 % der Fahrzeugbreite aufschlägt, wird der Effekt deutlich.
Bei diesem Szenario spielt Ko-TAG seinen Vorteil aus, da bereits frühzeitiger gewarnt werden kann und somit der Fahrer auf die Gefahrensituation hingewiesen werden kann. Auch bei Szenario 2 (Aufprall bei 75 %) ist durch Ko-TAG eine frühere Warnung als mit der Kamera möglich.

5.4 Simulation mit variablen Systemparametern (Szenario 1)

Die kamerabasierte Fußgängererkennung wurde in der Co-Simulation mit einem variierenden Systemparameter betrachtet. Die Simulation wurde anhand des Szenario 1 (Verdeckung, Aufprall Fahrzeugmitte und 5 km/h Fußgängergeschwindigkeit) durchgeführt. Um die Auswirkung des Bremszeitpunktes zu analysieren, wurde der Bremszeitpunkt im Funktionsalgorithmus bei jedem Simulationsdurchlauf verändert. Dadurch ergeben sich unterschiedliche Zeitpunkte, ab wann das Fahrzeug zu bremsen beginnt und somit unterschiedliche Kollisionspunkte und Kollisionsgeschwindigkeiten zwischen Fahrzeug und Fußgänger. Die Simulationsergebnisse befinden sich in Tabelle 5.4.

Systemparameter: Bremszeitpunkt TTC	Bremsw. [s]	y_PED [m]	x_VEH [m]	Kollision [s]	y_PED [m]	x_VEH [m]	v_VEH [m/s]	Δv_VEH [m/s]
0.5	-0.375	-0.51	-3	0.001	0.002	0	7.75	0.25
0.6	-0.470	-0.64	-3.76	0.007	0.009	0	7.34	0.66
0.7	-0.565	-0.77	-4.52	0.02	0.028	0	6.66	1.34
0.8	-0.66	-0.90	-5.28	0.05	0.07	0	5.68	2.32
0.9	-0.755	-1.03	-6.04	0.10	0.14	0	4.47	3.53
1.0	-0.944	-1.29	-7.55	0.28	0.38	-0.27	0	8
1.1	-1.039	-1.42	-8.31	0.37	0.51	-1.04	0	8
1.2	-1.134	-1.55	-9.07	0.47	0.64	-1.79	0	8

Tabelle 5.4: Simulationsergebnisse des kamerabasierten Fußgängerschutzes mit variierenden Systemparametern

Simulationsparameter:

```
a_veh_start: 0 m/s^2   % Fahrzeug bewegt sich mit konstanter Geschwindigkeit
v_veh_start: 28.8 km/h (t = -2 s) % Geschwindigkeit des Fahrzeuges 2 s vor Kollision
v_ped: 5 km/h % Fußgängergeschwindigkeit
y_ped_start: 2.64 m % Startposition des Fußgängers vom Fahrbahnmittelpunkt
x_veh_start: -18.33 m % Startposition des Fahrzeug-Nullpunktes vom Kollisionspunkt
x_veh_roadside: -3.33 m % x-Abstand des Verdeckungsfahrzeuges
y_veh_roadside: 2.97 m % y-Abstand des Verdeckungsfahrzeuges

Kamera:
Update-Rate des Funktionsalgorithmus: 0.94 s
Bremszeitpunkt: 0.5 s  -  1.2 s
```

In Abbildung 5.16 ist der x-Verlauf des Fahrzeuges dargestellt, dieser variiert je nach Bremszeitpunkt. Je später das Fahrzeug zum Stillstand kommt, desto größer wird die Zeit bis zum Aufprall des Fußgängers, dadurch verschiebt sich der laterale Kollisionspunkt in die linke Fahrzeughälfte. Die durchgezogenen vertikalen Linien stellen die Zeitpunkte der Bremsanforderung durch den Funktionsalgorithmus dar. Diese sind abhängig vom definierten Bremszeitpunkt, der Update-Rate des Funktionsalgorithmus und von der ermittelten Position des Fußgängers durch die Kamera. Deshalb weicht der Bremszeitpunkt von der definierten TTC ab. Zum Beispiel: Bei einem eingestellten Bremszeitpunkt von 1.2 s liegt dieser in der Simulation bei 1.13 s. Die strichlierten vertikalen Linien stellen den Kollisionspunkt bzw. den Stillstand des Fahrzeuges (1.0 s, 1.1 s, 1.2 s) dar.
Zusätzlich ist im Diagramm der Verlauf der TTC bei einem eingestellten Bremszeitpunkt von 0.5 s und 1.2 s dargestellt. Zwischen den beiden Verläufen ist gut erkennbar, dass sie sich zu Beginn nicht unterscheiden. Beim früheren Bremseingriff (1.2 s) erhöht sich die TTC und geht in Richtung unendlich, im Gegensatz zum späteren Bremseingriff (0.5 s).

Der Geschwindigkeitsverlauf des Fahrzeuges ist in Abbildung 5.17 dargestellt, der Geschwindigkeitsabbau kann direkt bei den strichlierten vertikalen Linien abgelesen werden. Die Linien stellen den Kollisionspunkt bzw. den Stillstand des Fahrzeuges dar und somit

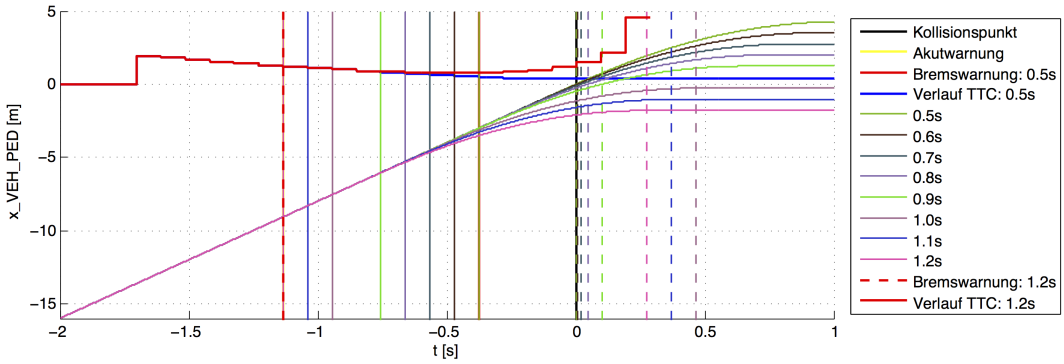

Abbildung 5.16: x-Abstand zwischen Fahrzeug und Fußgänger mit unterschiedlichen Bremszeitpunkten

die Kollisionsgeschwindigkeit. Bis zu einem Bremszeitpunkt größer gleich 1.0 s kommt es zu einer Vermeidung der Kollision. Bei späteren Bremseingriffen nimmt der Geschwindigkeitsabbau kontinuierlich ab und bei einem Bremseingriff unter 0.5 s kommt es zu keinem Abbau der Geschwindigkeit.

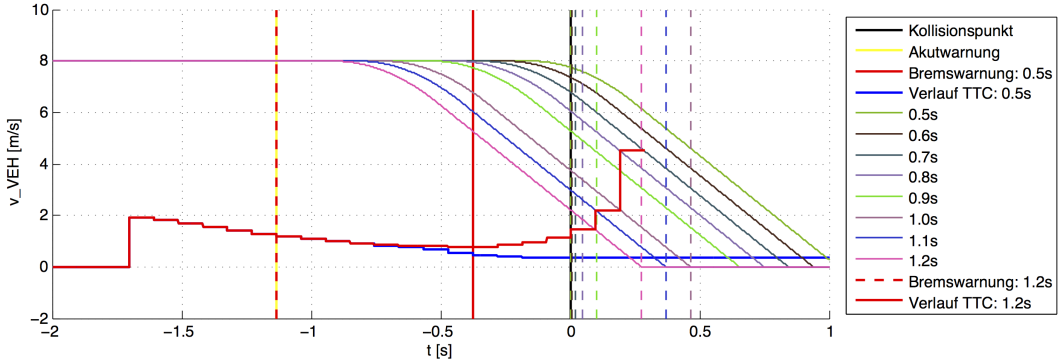

Abbildung 5.17: Geschwindigkeitsverlauf des Fahrzeuges mit unterschiedlichen Bremszeitpunkten

Der Geschwindigkeitsabbau in Abhängigkeit zum eingestellten Bremszeitpunkt ist in Abbildung 5.18 visualisiert. Der erwartete quadratische Verlauf des Geschwindigkeitsabbaus ist gut erkennbar. Bei der Simulation dieses Szenarios liegt der optimale Bremszeitpunkt bei einer TTC von 1.0 s, das Fahrzeug vermeidet die Kollision und kommt 27 cm vor dem Fußgänger zu stehen, ein früheres Bremsen wäre nicht notwendig, wenn man eine Kollision vermeiden möchte.

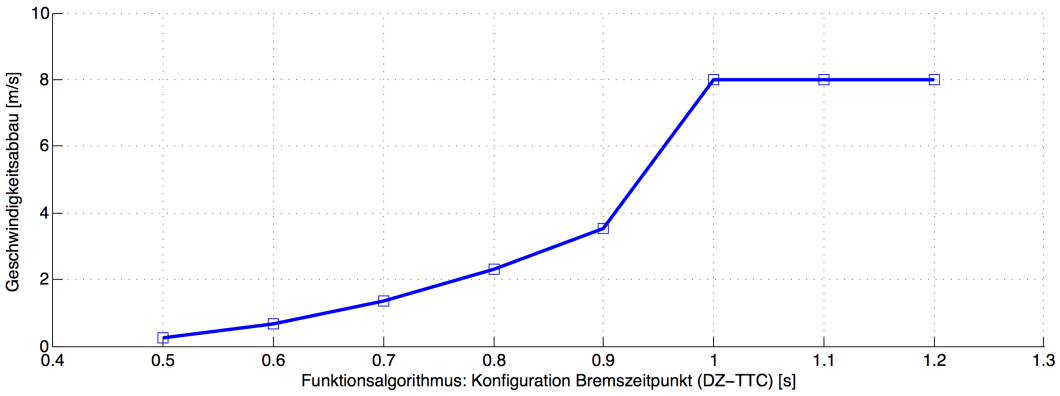

Abbildung 5.18: Geschwindigkeitsabbau abhängig vom Bremszeitpunkt

5.5 Interpretation Mess- und Simulationsergebnisse

Die Messungen am Prüfstand haben gezeigt, dass genaue und reproduzierbare Messungen nur mit einem Fahrroboter (Gas-/Bremsroboter) durchgeführt werden können, da es sonst zu Schwankungen in der Anfahrtsgeschwindigkeit und der Fahrzeugposition im Fahrschlauch kommt. Die unterschiedlichen Geschwindigkeiten haben variierende TTC-Berechnungen des Prüfstandes zur Folge, wodurch der Fußgänger zu unterschiedlichen Zeitpunkten gestartet wird und die Sichtbarkeit bei den einzelnen Messungen nicht gleich ist. Dies führt zu Abweichungen durch den Prüfstand und zu unterschiedlichen Ergebnissen aufgrund des Systemverhaltens der Sensoren. Dadurch ergeben sich neben dem Fehler durch das Modell zusätzliche Abweichungen bei der Parametrisierung der Sensoren für die Simulation. Zur Vermeidung dieses Problems muss ein Fahrroboter verwendet werden, um mehrere Fahrten mit den gleichen Bedingungen durchführen zu können. Dadurch sind die einzelnen Messfahrten identisch und bei der Vermessung der Sensoren können die Messergebnisse genauer gemittelt werden.

Vergleich Kamera - Ko-TAG

Bei der Kamera schwankt der Messfehler bei der Entfernungsmessung in x- und y-Richtung innerhalb eines Bereiches unabhängig vom Szenario, dies zeigen die Messergebnisse. Bei der Auswertung des y-Fehlers schwankt der gemittelte Fehler zwischen $\pm$ 10 cm zu den einzelnen Szenarien. Betrachtet man die Simulationsergebnisse der Szenarien, so schwankt auch dort die Abweichung zwischen der realen Messung und der Simulation innerhalb dieses Bereiches. Eine Ausnahme stellt Szenario 3 (Aufprall 25 % mit 8 km/h) dar. Die Messwerte bei diesem Szenario wurden in die Kameraauswertung nicht mit aufgenommen, da sie zu stark von den anderen Messungen abweichen, deshalb kommt es bei diesem Szenario auch zu den größten Abweichungen zwischen Simulation und Messung.
Die Entfernungsmessung in x-Richtung weicht im Gegensatz zur y-Messung mehr von der

tatsächlichen Position ab. Abweichungen von bis zu 200 cm treten bei Entfernungen über 14 m auf. Bei Entfernungen unter 14 m wird der Fehler kleiner und pendelt sich bei 50 cm ein. Die Abweichungen zwischen dem gemittelten Fehler und den einzelnen Messungen bewegen sich in einem Bereich von 50 cm. Auch in der Simulation liegt der Fehler zur tatsächlichen Position in diesem Bereich. Die Gültigkeit der Modellierung ist somit auch nur in dem Bereich gegeben, in dem die Genauigkeit der Kamera liegt.
Verglichen mit der Kamera kommt es bei den Messungen mit Ko-TAG zu größeren Abweichungen bei der y-Messung, dafür ist die Entfernungsmessung in x-Richtung genauer. Systembedingt war dieses Verhalten zu erwarten, da die Entfernungsmessung über Time-of-Flight robuster ist als die Winkelmessung (Direction-of-Arrival). Einflussgrößen wie z. B. eine Mehrfachausbreitung der Wellen wirken sich stärker aus, da die Phasendifferenz einer Welle zwischen den Antennen ausgewertet wird und somit empfindlicher ist als die Messung, wann ein Paket versendet und angekommen ist. Prinzipiell spricht nichts gegen eine genauere y-Messung. Auch bei der Entfernungsmessung in x-Richtung kommt es beim Ko-TAG-Sensor zu Abweichungen von bis zu 300 cm, wenn sich der Fußgänger mehr als 15 m vom Fahrzeug entfernt befindet. Dies hat bei den durchgeführten Szenarien aber keine Auswirkung, da der Fußgänger bei dieser Entfernung noch zu weit vom Fahrschlauch entfernt ist und es dadurch zu keiner kritischen Situation kommt. Beim verwendeten Ko-TAG-Sensor ist zu bedenken, dass sich dieser in einem frühen Stadium der Entwicklung befindet, verglichen mit dem Kamerasystem, das bereits seriennahe ist.
Mit Ko-TAG ist die Messung der lateralen Position ungenauer. Dies hat den Nachteil, dass die Bestimmung des Warnzeitpunktes ungenauer ist, da zum Zeitpunkt der Warnung hauptsächlich entscheidend ist, wie weit sich der Fußgänger vom Fahrschlauch entfernt befindet. Der Bremszeitpunkt lässt sich dafür aber exakter bestimmen, da die Entfernung in x-Richtung genauer ist und sich der Fußgänger zu diesem Zeitpunkt bereits im Fahrschlauch befindet. Der Fehler in y-Richtung spielt somit keine große Rolle.
Bei allen Messungen mit Ko-TAG ist gut erkennbar, dass unabhängig vom Szenario die y-Position des Fußgängers bis 8 m vor der Kollision zu weit im Fahrschlauch bestimmt wird, danach wird der Fehler kleiner $\pm$ 20 cm. Dieser Fehler kann mehrere Ursachen haben und kann nicht nur auf die Verdeckung zurückgeschlossen werden, da dieser Fehler auch bei freier Sichtverbindung auftritt. Eine Ursache dafür könnte der Prüfstand sein, die genaue Ursache konnte im Zuge dieser Messungen nicht eruiert werden.

Funktionsalgorithmus

Der Funktionsalgorithmus funktioniert beim Fußgängerschutz mit Ko-TAG am Prüfstand wie definiert. Der Algorithmus löst wie implementiert aus, wenn sich der Fußgänger in den definierten Warn- und Bremskorridoren befindet. Auch bei der Simulation mit der Kamera und Ko-TAG weist der Algorithmus das gewünschte Verhalten auf.

Kapitel 6

Zusammenfassung und Ausblick

In dieser Masterarbeit wird die Co-Simulation zur Entwicklung und Analyse des präventiven Fußgängerschutzes in der frühen Phase betrachtet. Zur Fußgängererkennung werden der kooperativen Sensor Ko-TAG und eine Monokamera verwendet. Die Arbeit befasst sich mit: Vermessung, Modellierung, Parametrisierung und Simulation des präventiven Fußgängerschutzes.
Die Parametrisierung des Kamera- und Ko-TAG-Modelles erfolgte durch Messungen am Fußgängerprüfstand. Die wichtigsten Parameter zur Beschreibung der Sensormodelle sind die Identifikationszeit, die Dauer, bis der Sensor Daten vom Fußgänger liefert, und der Messfehler bei der Positionsbestimmung. Die Messungen am Prüfstand haben gezeigt, dass die Messfehler bei der Positionsbestimmung unabhängig vom Szenario den gleichen Verlauf aufweisen und innerhalb eines Fehlerbereiches liegen. Aus diesem Grund werden zum Parametrisieren der Modelle die Parameter in Look-Up-Tabellen gespeichert. In der Simulation greifen die Sensormodelle bei der Positionsbestimmung des Fußgängers und der Identifikationszeit der Kamera auf die Look-Up-Tabellen zurück. Mit dieser Methode zur Parametrisierung von Sensoren liegt die Genauigkeit der Simulation innerhalb der Grenzen, in denen auch die Sensoren schwanken. Treten z. B. bei der Positionsbestimmung zwischen den einzelnen Szenarien Unterschiede von 50 cm bei der y-Messung auf, so wird auch in der Simulation der Fehler innerhalb dieses Bereiches liegen. Betrachtet man die Gegenüberstellung der Mess- und Simulationsergebnisse in Kapitel 5.3.4 und im Anhang A, so liegen die Ergebnisse der Simulationen innerhalb des bestimmten Messfehlers.
Bei den Messwerten des kooperativen Sensor Ko-TAG ist zu beachten, dass der Sensor in einem frühen Entwicklungsstadium steckt und prototypisch gefertigt wurde. Verglichen mit der Monokamera kann dadurch nur die derzeitige Genauigkeit des Sensors betrachtet werden und keine Aussage gemacht werden, wie genau der Sensor prinzipiell ist. Der Sensor zeigt aber großes Potenzial bei der frühzeitigen Erkennung von Fußgängern oder auch bei Radfahrern. Besonders gilt dies für Situationen, wenn Fußgänger aus der Verdeckung

kommen und sich mit hoher Geschwindigkeit in Richtung des Fahrschlauches bewegen. In diesem Szenario ist mit einer Kamera keine Warnung mehr möglich, da der Fußgänger erst viel zu spät detektiert wird und bereits ein Bremseingriff nötig ist. Mit Ko-TAG ist hingegen eine frühere Warnung möglich, da der Transponder am Fußgänger seinen Zustand an das Fahrzeug senden kann und die Positionsbestimmung möglich ist. Um genauere Ergebnisse bei einem kooperativen System wie Ko-TAG zu erhalten, hat sich gezeigt, dass die Simulation über ein einfaches Modell nur bedingt realisierbar ist. Für eine genauere Simulation müssen komplexere Modelle verwendet werden, die auch die Mehrwegeausbreitung berücksichtigen, welche abhängig von der Verdeckung ist.
Die Gesamtsimulation des präventiven Fußgängerschutzes kann für weitere Analysen verwendet werden, um z. B. Auswirkungen einzelner Parametern zu untersuchen. Es ist aber nicht möglich, Ergebnisse wie den Kollisionspunkt zentimetergenau mit einer Messfahrt am Fußgängerprüfstand zu vergleichen. Das Hauptproblem liegt darin, dass zu viele Parameter bei einer Messfahrt variieren können. Um diese Probleme zu lösen, muss ein Fahrroboter verwendet werden und mehrere Fahrten mit den gleichen Bedingungen durchgeführt werden. Dadurch werden genauere Messdaten gewonnen, mit Hilfe derer die Modelle besser parametrisiert werden können.
Durch die Verwendung der Co-Simulationsumgebung ICOS ist es möglich, die Simulation dahingehend zu erweitern, um die Auswirkungen auf den Fußgänger nach dem Kollisionszeitpunkt zu analysieren. Ab dem Kollisionszeitpunkt können zusätzliche Modelle und Simulationstools verwendet werden, mit denen das Crashverhalten simuliert werden kann.

Zur Verbesserung der Simulationsergebnisse müssen einerseits die Modelle durch reproduzierbarere Messfahrten besser parametrisiert werden, andererseits muss das Bremsmodell für unterschiedliche Straßenzustände weiterentwickelt werden und bei der Analyse kooperativer Sensoren muss ein detaillierteres Modell verwendet werden. Darüber hinaus wird derzeit bei der Simulation nur die Systemreaktion bewertet, die Auswirkungen auf die/den FahrerIn durch die Warnung wird nicht berücksichtigt. Um auch auf die Reaktion der/des FahrerIn zu reagieren, müsste im nächsten Schritt zusätzlich ein Fahrermodell verwendet werden, das gegebenenfalls eine Bremsung auslösen kann.

Literaturverzeichnis

[Amu09] Forschungsprojekt Amulett. *Aktive mobile Unfallvermeidung und Unfallfolgenminderung durch kooperative Erfassungs- und Trackingtechnologie.* http://www.projekt-amulett.de, 2013-02-21.

[Blo12] T. Blochwitz. *Functional Mockup Interface 2.0: The Standard for Tool independent Exchange of Simulation Models.* Proceedings of the 9th International MODELICA Conference, München, 2012.

[BZWB12] Martin Benedikt, Josef Zehetner, Daniel Watzenig, Jost Bernasch. *Moderne Kopplungsmethoden – Ist Co-Simulation beherrschbar?* NAFEMS - Zeitschrift für numerische Simulationsmethoden und angrenzende Gebiete, Nr. 2/2012, Bernau, 2012.

[BMW10] BMW Group. *Connected Drive: Fahrerassistenz der Zukunft – mein Schutzengel fährt mit.* BMW Presse-Mitteilung, 2010.

[Chr09] Mohcine, Chraibi. *System gewöhnlicher Differentialgleichungen zur Beschreibung von Fußgängerdynamik.* Technische Universität Hamburg, 2009.

[Dom12] Christian Domsch. *Leistungssteigerung präventiver Schutzsysteme.* 5. Tagung Fahrerassistenz, Fahrzeugsicherheit, 2012.

[Eag13] Elektrobit Automotive GmbH *Aufzeichnungsumgebung Kamerabasierte Fahrerassistenz.* http://automotive.elektrobit.com/home/driver-assistance-software/eb-assist-adtf.html, 2013-02-15.

[Erb09] Christian Erbsmehl. *Simulation of real crashes as a method for estimating the potential benefits of advanced safety technologies, ESV Paper No. 09-0162.* Verkehrsunfallforschung an der TU Dresden (VUFO) GmbH, ESV Konferenz, 2009.

[Fen10] Liu Feng. *Objektverfolgung durch Fusion von Radar- und Monokameradaten auf Merkmalsebene für zukünftige Fahrerassistenzsysteme.* KIT Scientific Publishing, 2010.

[GKL06] M. Geimer, T. Krueger, P. Linsel. *Co-Simulation, gekoppelte Simulation oder Simulatorkopplung, „Co-Simulation, coupled simulation or simulator coupling“.* Zeitschrift für Fluidtechnik, vol. 50, 2006.

[Hei12] Sascha Heinrichs-Bartscher. *Die Anforderungen von Euro-NCAP an die Umfeldsensorik.* http://www.all-electronics.de/texte/anzeigen/48675/AEB-etc-bald-in-allen-Fahrzeugen, 2013-02-07.

[Kld13] Horst Klöden. *Fahrzeugsicherheit durch kooperative Sensorik.* Technische Universität Chemnitz, 2013.

[Kof13] Forschungsinitiative Ko-FAS. *Forschungsinitiative bestehend aus drei Verbundprojekten: Ko-TAG, Ko-PER und Ko-KOMP.* http://ko-fas.de, 2013-04-10.

[KSW10] Michael Karner, Christian Steger, Reinhold Weiß. *A Cross Domain Co-Simulation Platform for the Efficient Analysis of Mechatronic Systems, SAE Paper No. 2010-01-0239.* SAE World Congress Detroit, 2010.

[Mat13] The Math Works Inc. *Matlab, Simulink. Rechen-/ Simulationsprogramm.* http://www.mathworks.de/products/simulink, 2013-02-13.

[Mic13] Microchip Technology Inc. *32bit Mikroprozessor Familie.* http://www.microchip.com/pagehandler/en-us/family/32bit/, 2012-12-10.

[Pra10] Prat Alvora Catala. *Sensordatenfusion und Bildverarbeitung zur Objekt- und Gefahrenerkennung.* Technische Universität Carolo-Wilhelmina zu Braunschweig, 2010.

[Rei10] Konrad Reif. *Fahrstabilisierungssysteme und Fahrerassistenzsysteme.* Springer Fachmedien Wiesbaden GmbH, 2010.

[Rei12] Konrad Reif. *Automobilelektronik, Eine Einführung für Ingenieure.* Springer Fachmedien Wiesbaden GmbH, 2012.

[RSMB09] Rasshofer Ralph H., Daniel Schwarz, Christian Morhart, Erwin Biebl. *Cooperative Sensor Technology for Preventive Vulnerable Road User Protection.* German Microwave Conference, 2009.

[Sch12] Daniel Schwarz. *Extension of Car-to-X-Communication by Radiolocation Techniques.* ATZelektronik woldwide, 05/2012.

[Sra11] Stefan Schramm. *Methode zur Berechnung der Feldeffektivität integraler Fußgängerschutzsysteme.* Technischen Universität München, 2011.

[SKW13] Markus Schratter, Michael Karner, Peter Wimmer, Daniel Watzenig, Christian Gruber. *A Co-Simulation Based Approach for the Validation of Integrated Safety Systems.* SAE World Congress Detroit, 2013.

[TBS09] Nils Tiemann, Wolfgang Brand, Dieter Schramm. *Predictive Pedestrian protection - Sensor Requirements and Risk Assessment.* Robert Bosch GmbH, Universität Duisburg-Essen, 2009.

[Tie12] Nils Tiemann. *Ein Beitrag zur Situationsanalyse im vorausschauenden Fußgängerschutz.* Universität Duisburg-Essen, 2012.

[Vif13] Kompetenzzentrum - Das virtuelle Fahrzeug Forschungsgesellschaft mbH *Co-Simulationsumgebung ICOS.* http://vif.tugraz.at/forschung/ee-software/co-simulation/icos, 2013-01-07.

[Wat08] Forschungsprojekt Watch-Over. *eSafety Co-operative Systems for Road Transport.* http://www.watchover-eu.org, 2013-04-10.

[WHW12] Wolf, Winner, Hakuli. *HandbuchFahrerassistenzsysteme, Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort.* Springer Fachmedien Wiesbaden GmbH, 2012.

[WZ12] Wenpu Lu, Zehetner Josef. *Stability Analysis of a Two-Voltage Vehicle Electrical System Based on Co-Simulation, SAE Paper No. 2012-01-0012.* SAE World Congress Detroit, 2012.

[Zeh11] Zehetner Josef. *Co-Simulation Based Evaluation of an Energy Management System for Hybrid Electric Vehicles: Presentation, IQPC Thermal Management for EV/HEV.* Darmstadt, 2011.

Anhang A

Mess- und Simulationsergebnisse

A.1 y-/x-Verlauf des Fußgängers und des Fahrzeuges

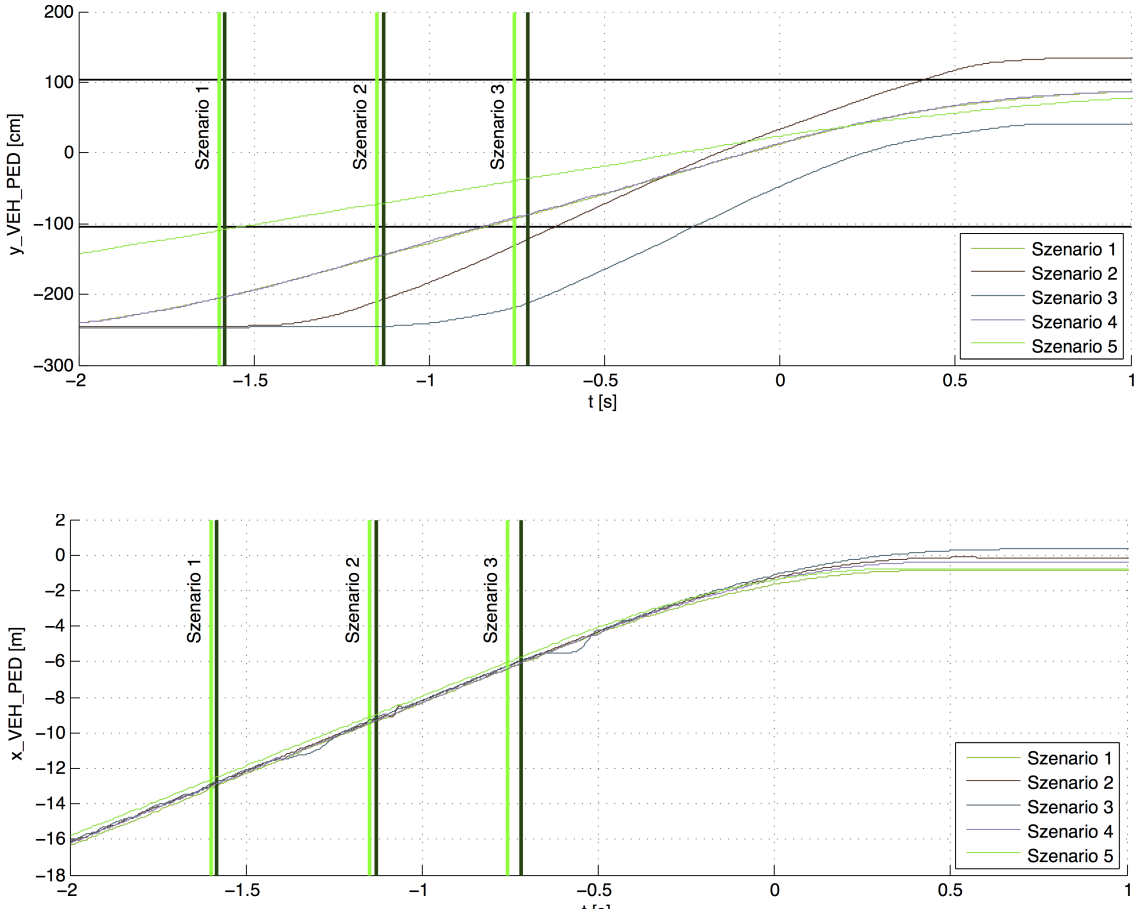

Abbildung A.1: x-Verlauf des Fahrzeuges der gemessenen Szenarien

A.2 Szenario 2

Simulationsparameter

```
a_veh_start: -0.2 m/s^2   %Fahrzeug bewegt sich nicht mit konstanter Geschwindigkeit
v_veh_start: 29.2 km/h (t = -2 s) % Geschwindigkeit des Fahrzeuges 2 s vor Kollision
v_ped: 7.74 km/h % Fußgängergeschwindigkeit
y_ped_start: 4.10 m % Startposition des Fußgängers vom Fahrbahnmittelpunkt
x_veh_start: -18.83 m % Startposition des Fahrzeug-Nullpunktes vom Kollisionspunkt
x_veh_roadside: -3.33 m % x-Abstand des Verdeckungsfahrzeuges
y_veh_roadside: 2.97 m % y-Abstand des Verdeckungsfahrzeuges

Kamera:
Update-Rate Funktionsalgorithmus: 0.094 s
Bremszeitpunkt in der Simulation:  1.15 s   % Auslösung durch Ko-TAG bei der Messfahrt

Ko-TAG:
Update-Rate Funktionsalgorithmus: 0.05 s
```

Kamerabasierte Fußgängererkennung

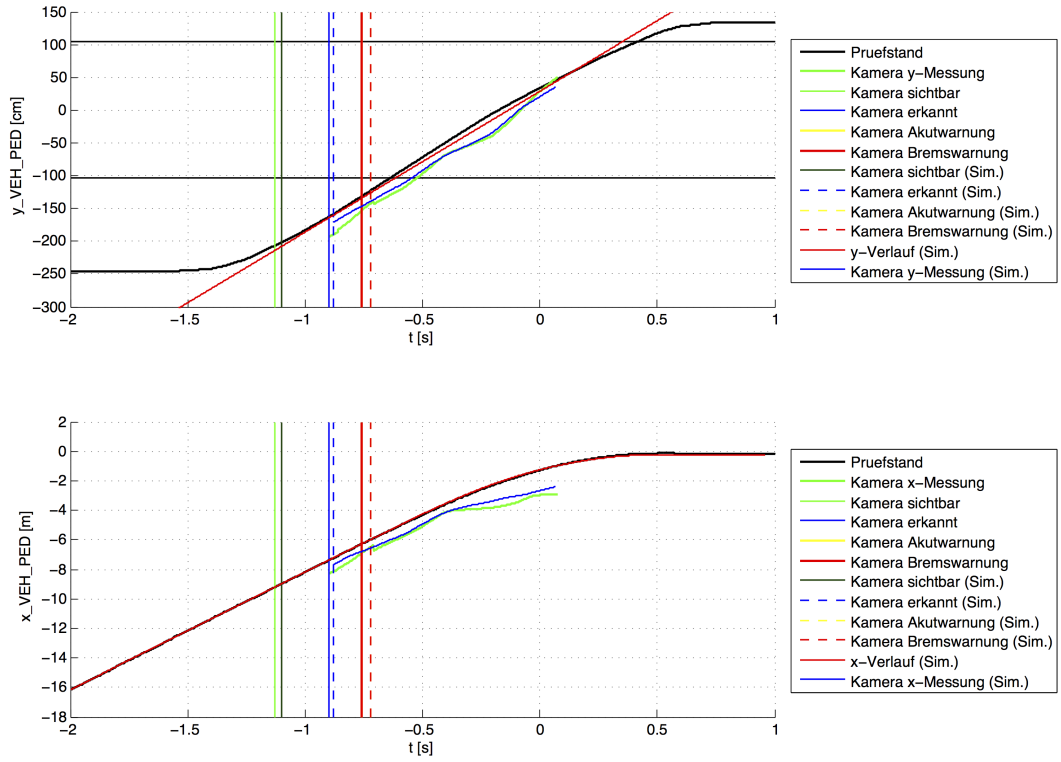

Abbildung A.2: y/x-Verlauf des Fußgängers mit der Kamera, Messung und Simulation

Nr.	sichtbar [s]	y_PED [m]	x_VEH [m]	erkannt y_PED [m]	erkannt x_VEH [m]	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
111156	-1.10	-2.07	-9.13	-1.96	-8.58	-0.76	-1.31	-6.33	-0.76	-1.31	-6.33
111329	-1.12	-2.07	-9.19	-1.24	-5.77	-0.71	-1.24	-5.77	-0.71	-1.24	-5.77
111457	-1.11	-2.07	-9.23	-1.54	-7.13	-0.77	-1.34	-6.43	-0.77	-1.34	-6.43
111951	-1.17	-2.07	-9.61	-1.73	-8.03	-0.79	-1.34	-6.58	-0.79	-1.34	-6.58
MW	-1.13	-2.07	-9.29	-1.62	-7.38	-0.76	-1.31	-6.28	-0.76	-1.32	-6.29
Sim.	-1.10	-2.08	-8.97	-1.60	-7.22	-0.71	-1.24	-5.92	-0.71	-1.24	-5.92
Diff.	0.03	-0.01	0.32	0.02	0.16	0.05	0.07	0.36	0.05	0.08	0.37

Tabelle A.1: Mess- und Simulationsergebnisse der Kamera

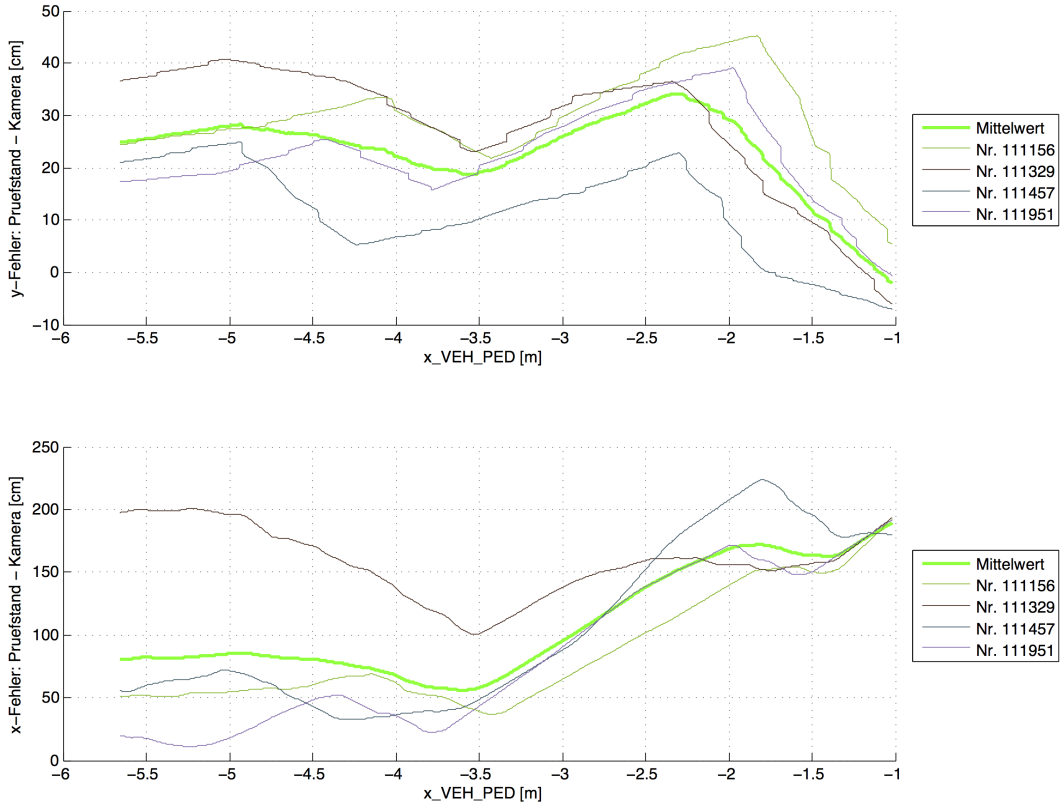

Abbildung A.3: y/x-Verlauf der Fehlerabweichung der Kamera

Ko-TAG-basierte Fußgängererkennung

Nr.	sichtbar [s]	y_PED [m]	x_VEH [m]	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
111156	-1.12	-2.11	-9.20	-1.04	-1.97	-8.64	-0.98	-1.85	-8.08
111329	-1.15	-2.10	-9.28	-1.05	-1.96	-8.49	-0.99	-1.83	-8.05
111457	-1.12	-2.10	-9.32	-1.13	-2.09	-9.28	-0.95	-1.72	-7.83
111951	-1.19	-2.10	-9.72	-1.12	-1.96	-9.12	-0.98	-1.73	-8.02
MW	-1.15	-2.09	-9.38	-1.09	-1.99	-8.89	-0.97	-1.82	-8.02
Sim.	-1.11	-2.10	-9.08	-1.13	-2.11	-9.28	-0.98	-1.84	-8.09
Diff.	0.04	-0.01	0.30	-0.04	-0.12	-0.37	-0.01	-0.02	-0.07

Tabelle A.2: Mess- und Simulationsergebnisse des Ko-TAG-Sensors

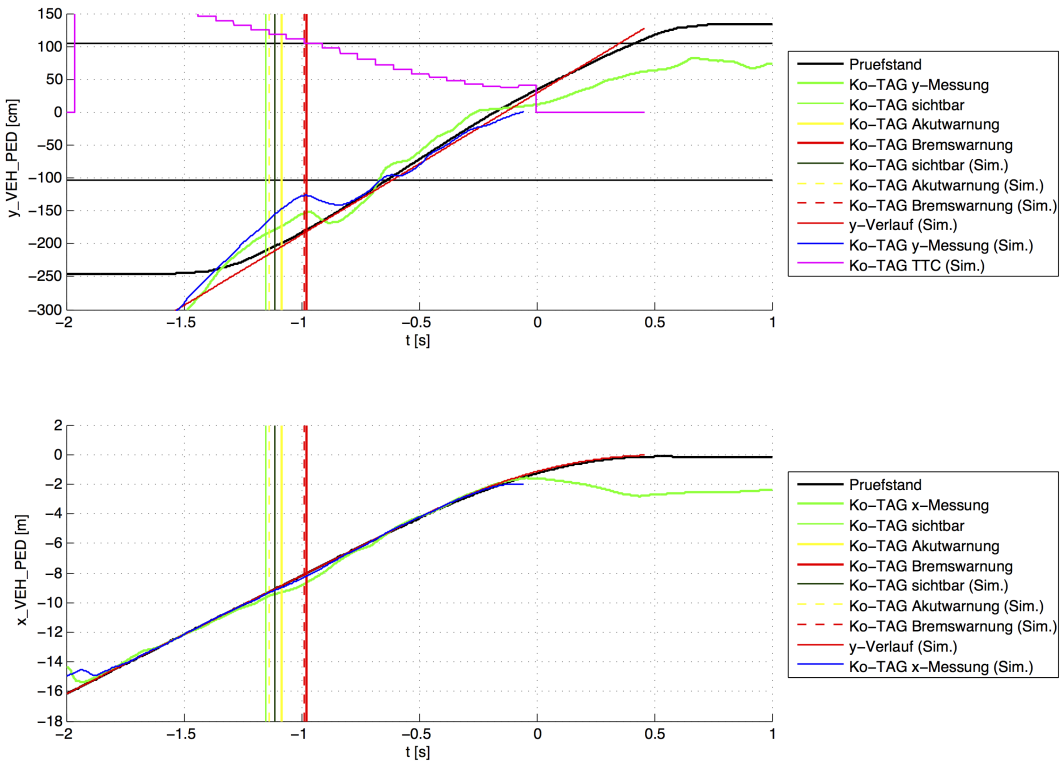

Abbildung A.4: y/x-Verlauf des Fußgängers mit Ko-TAG, Messung und Simulation

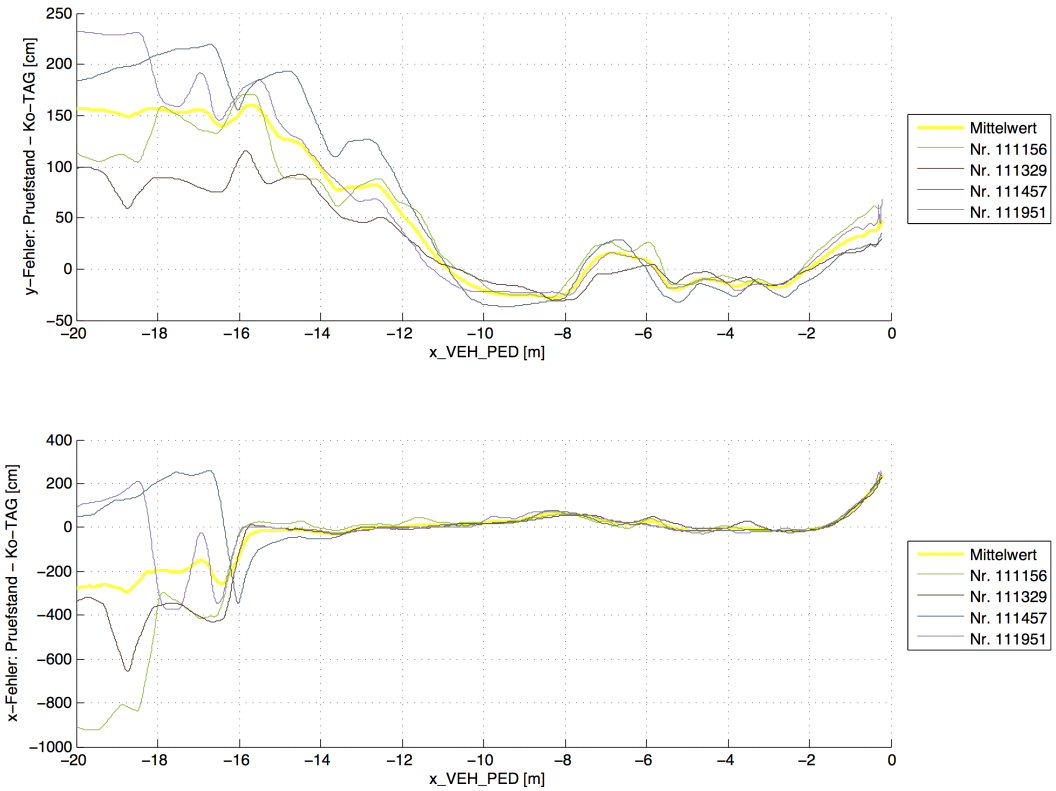

Abbildung A.5: y/x-Verlauf der Fehlerabweichung bei Ko-TAG

A.3 Szenario 3

Simulationsparameter

```
a_veh_start: -0.2 m/s^2   %Fahrzeug bewegt sich nicht mit konstanter Geschwindigkeit
v_veh_start: 29.2 km/h (t = -2 s) % Geschwindigkeit des Fahrzeuges 2 s vor Kollision
v_ped: 8.17 km/h % Fußgängergeschwindigkeit
y_ped_start: 5.12 m % Startposition des Fußgängers vom Fahrbahnmittelpunkt
x_veh_start: -18.83 m % Startposition des Fahrzeug-Nullpunktes vom Kollisionspunkt
x_veh_roadside: -3.33 m % x-Abstand des Verdeckungsfahrzeuges
y_veh_roadside: 2.97 m % y-Abstand des Verdeckungsfahrzeuges

Kamera:
Update-Rate Funktionsalgorithmus: 0.094 s
Bremszeitpunkt in der Simulation:  1.20 s   % Auslösung durch Ko-TAG bei der Messfahrt

Ko-TAG:
Update-Rate Funktionsalgorithmus: 0.05 s
```

Kamerabasierte Fußgängererkennung

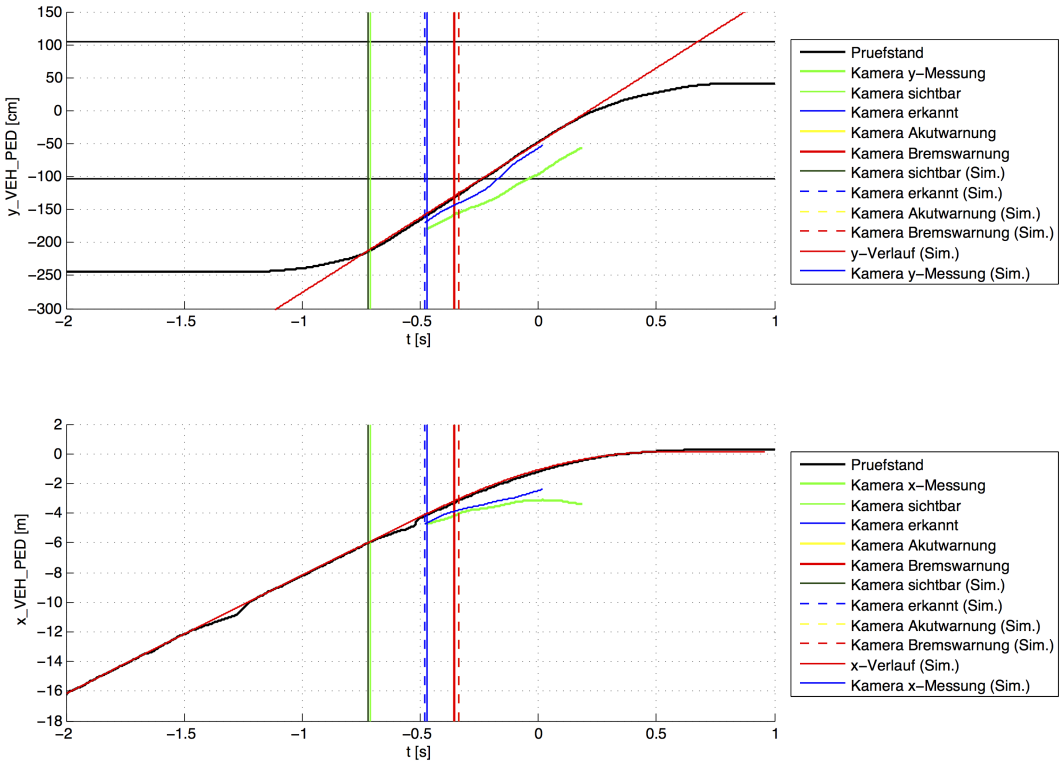

Abbildung A.6: y/x-Verlauf des Fußgängers mit der Kamera, Messung und Simulation

Nr.	sichtbar [s]	y_PED [m]	x_VEH [m]	erkannt y_PED [m]	erkannt x_VEH [m]	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
105515	-0.70	-2.13	-5.85	-1.70	-4.37	-0.34	-1.27	-3.12	-0.34	-1.27	-3.12
105658	-0.73	-2.13	-5.88	-0.41	-0.75	-0.02	-0.41	-0.77	-0.02	-0.41	-0.77
105825	-0.68	-2.13	-5.96	-1.38	-3.49	-0.35	-1.38	-3.50	-0.35	-1.38	-3.50
110612	-0.75	-2.12	-6.07	-1.70	-7.16	-0.38	-1.31	-3.24	-0.38	-1.31	-3.24
MW	-0.72	-2.13	-5.95	-1.59	-4.01	-0.36	-1.32	-3.31	-0.36	-1.32	-3.31
Sim.	-0.73	-2.14	-6.01	-1.58	-4.06	-0.34	-1.26	-3.06	-0.34	-1.26	-3.06
Diff.	-0.01	-0.01	-0.06	0.01	-0.05	0.02	-0.06	-0.25	0.02	-0.06	-0.25

Tabelle A.3: Mess- und Simulationsergebnisse der Kamera

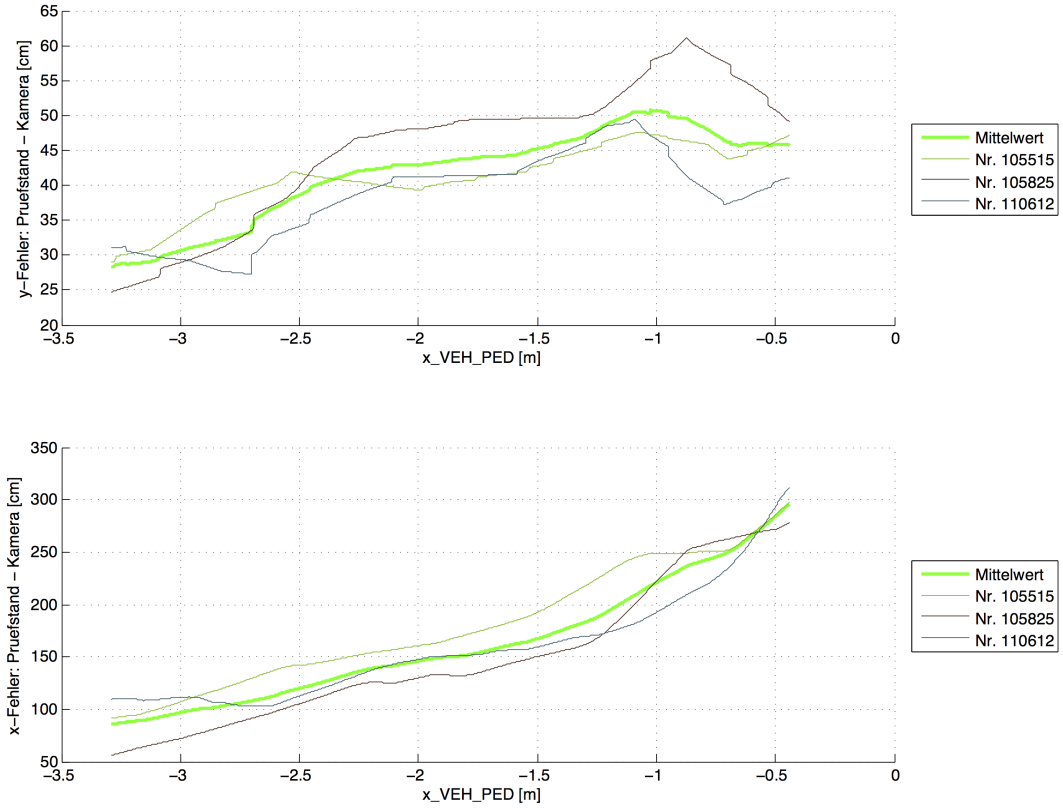

Abbildung A.7: y/x-Verlauf der Fehlerabweichung der Kamera

Ko-TAG-basierte Fußgängererkennung

Nr.	sichtbar [s]	y_PED [m]	x_VEH [m]	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
111156	-1.12	-2.11	-9.20	-1.04	-1.97	-8.64	-0.98	-1.85	-8.08
111329	-1.15	-2.10	-9.28	-1.05	-1.96	-8.49	-0.99	-1.83	-8.05
111457	-1.12	-2.10	-9.32	-1.13	-2.09	-9.28	-0.95	-1.72	-7.83
111951	-1.19	-2.10	-9.72	-1.12	-1.96	-9.12	-0.98	-1.73	-8.02
MW	-1.15	-2.09	-9.38	-1.09	-1.99	-8.89	-0.97	-1.82	-8.02
Sim.	-1.11	-2.10	-9.08	-1.13	-2.11	-9.28	-0.98	-1.84	-8.09
Diff.	0.04	-0.01	0.30	-0.04	-0.12	-0.37	-0.01	-0.02	-0.07

Tabelle A.4: Mess- und Simulationsergebnisse des Ko-TAG-Sensors

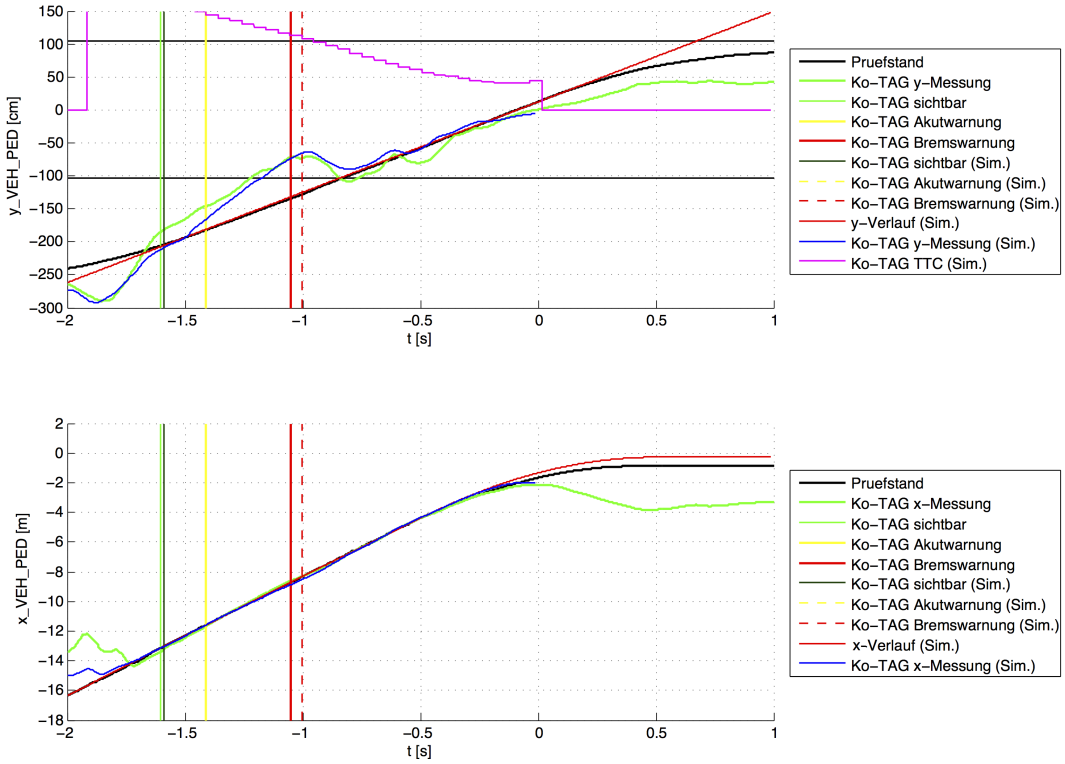

Abbildung A.8: y/x-Verlauf des Fußgängers mit Ko-TAG, Messung und Simulation

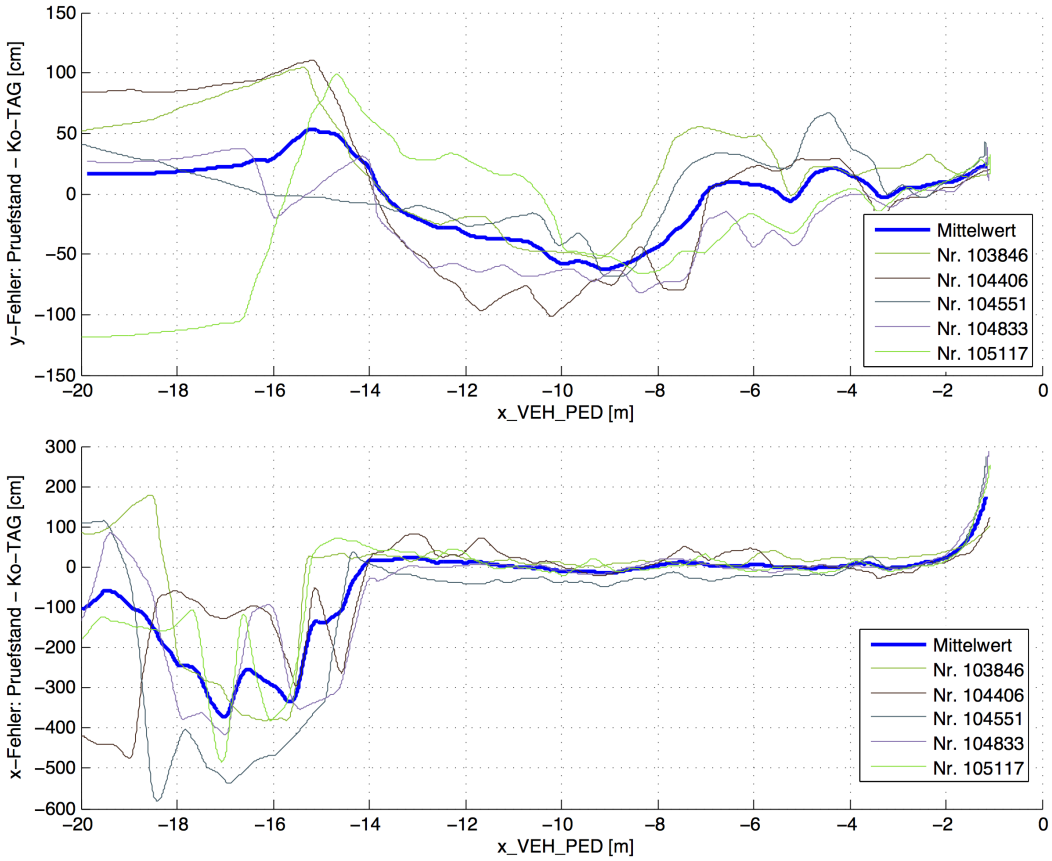

Abbildung A.9: y/x-Verlauf der Fehlerabweichung bei Ko-TAG

A.4 Szenario 4

Simulationsparameter

```
a_veh_start: -0.29 m/s^2   %Fahrzeug bewegt sich nicht mit konstanter Geschwindigkeit
v_veh_start: 29.5 km/h (t = -2 s) % Geschwindigkeit des Fahrzeuges 2 s vor Kollision
v_ped: 4.93 km/h % Fußgängergeschwindigkeit
y_ped_start: 2.64 m % Startposition des Fußgängers vom Fahrbahnmittelpunkt
x_veh_start: -18.83 m % Startposition des Fahrzeug-Nullpunktes vom Kollisionspunkt
x_veh_roadside: -3.33 m % x-Abstand des Verdeckungsfahrzeuges
y_veh_roadside: -30 m % y-Abstand des Verdeckungsfahrzeuges

Kamera:
Update-Rate Funktionsalgorithmus: 0.094 s
Bremszeitpunkt in der Simulation:  1.13 s   % Auslösung durch Ko-TAG bei der Messfahrt

Ko-TAG:
Update-Rate Funktionsalgorithmus: 0.05 s
```

Kamerabasierte Fußgängererkennung

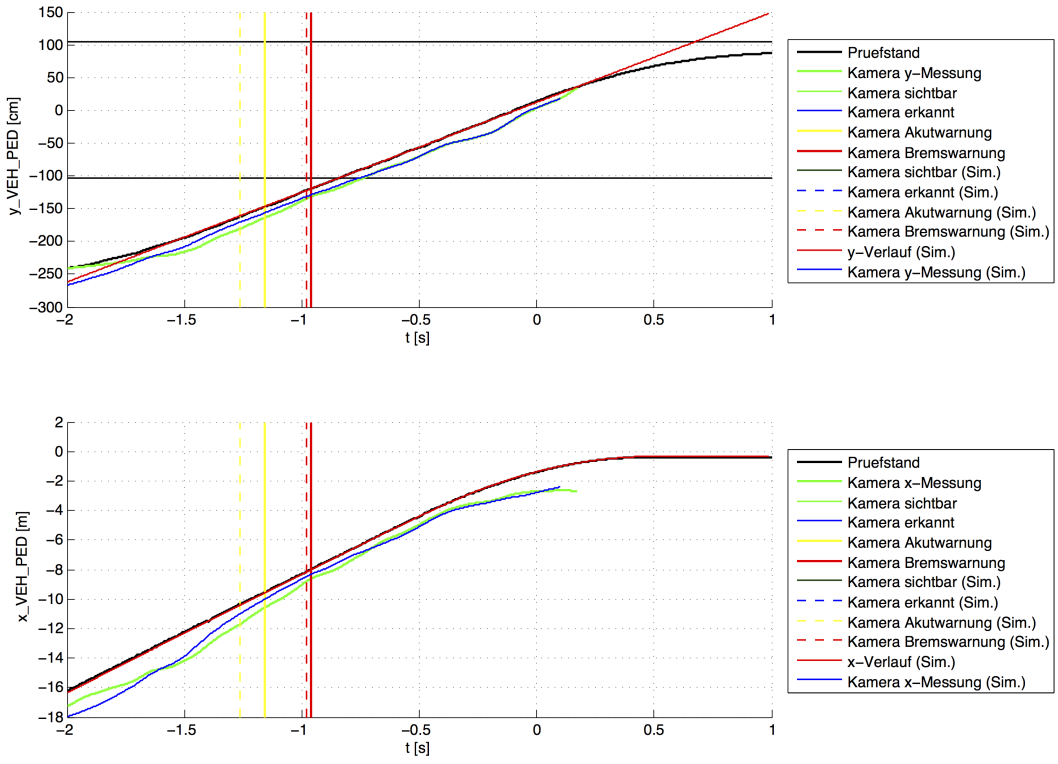

Abbildung A.10: y/x-Verlauf des Fußgängers mit der Kamera, Messung und Simulation

Nr.	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
115134	-1.24	-1.55	-10.27	-1.04	-1.27	-8.56
115656	-1.17	-1.46	-9.57	-0.98	-1.19	-8.01
115821	-1.16	-1.49	-9.48	-0.96	-1.22	-7.97
115953	-1.06	-1.38	-8.92	-0.87	-1.10	-7.37
MW	-1.16	-1.47	-9.56	-0.96	-1.19	-7.97
Sim.	-1.20	-1.53	-9.96	-0.94	-1.16	-7.83
Diff.	-0.04	-0.06	-0.40	0.02	0.03	0.14

Tabelle A.5: Mess- und Simulationsergebnisse der Kamera

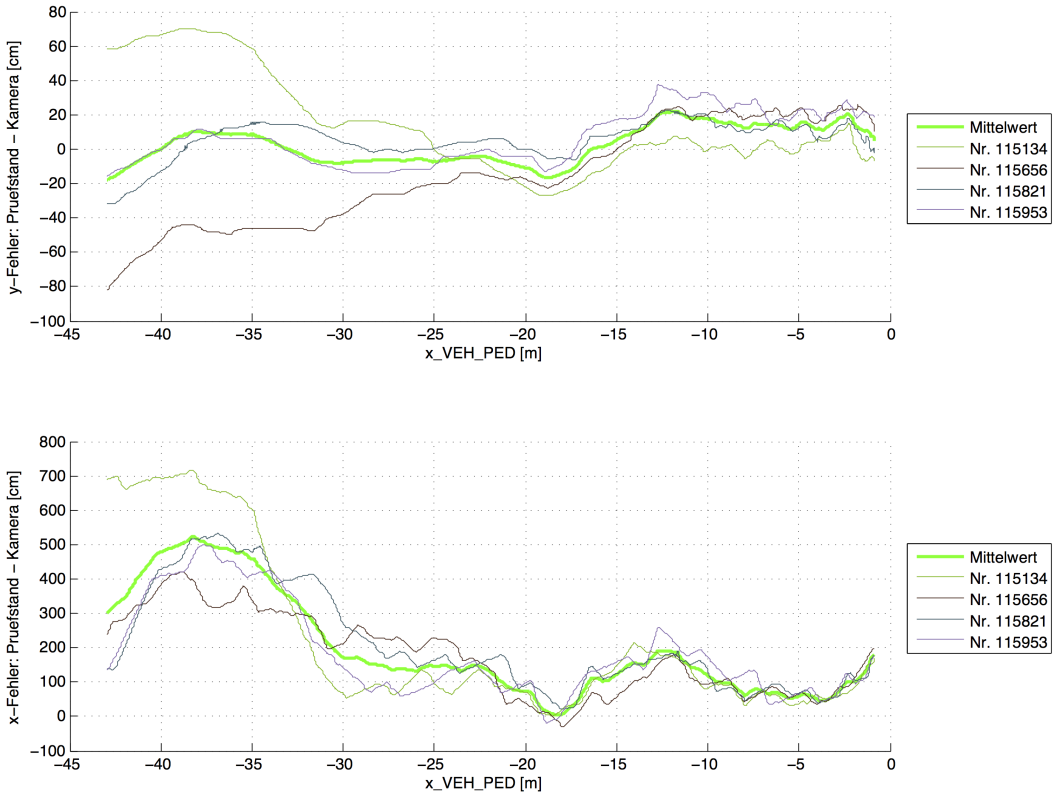

Abbildung A.11: y/x-Verlauf der Fehlerabweichung der Kamera

Ko-TAG-basierte Fußgängererkennung

Nr.	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
115134	-1.38	-1.76	-11.49	-1.10	-1.36	-9.14
115656	-1.39	-1.77	-11.21	-1.09	-1.36	-8.89
115821	-1.36	-1.79	-11.12	-1.06	-1.37	-8.67
115953	-1.45	-1.92	-11.98	-0.96	-1.23	-8.09
MW	-1.40	-1.81	-11.45	-1.05	-1.33	-8.70
Sim.	-1.42	-1.82	-11.61	-1.06	-1.32	-8.79
Diff.	-0.02	-0.01	-0.16	-0.01	0.01	-0.09

Tabelle A.6: Mess- und Simulationsergebnisse des Ko-TAG-Sensors

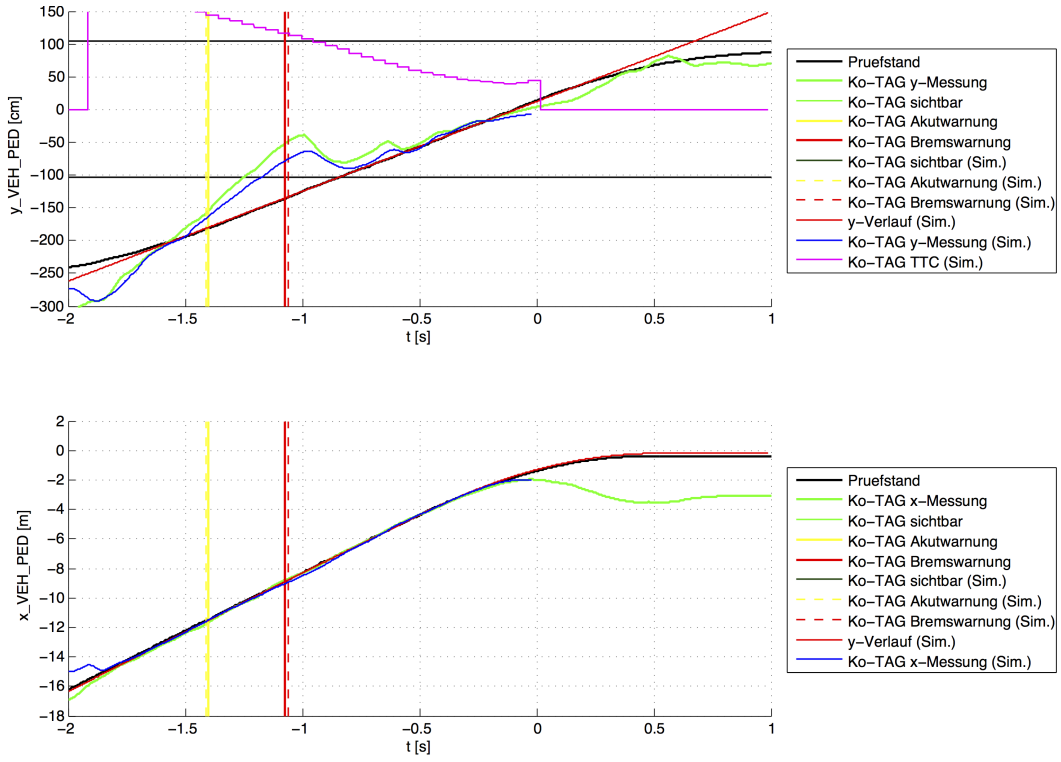

Abbildung A.12: y/x-Verlauf des Fußgängers mit Ko-TAG, Messung und Simulation

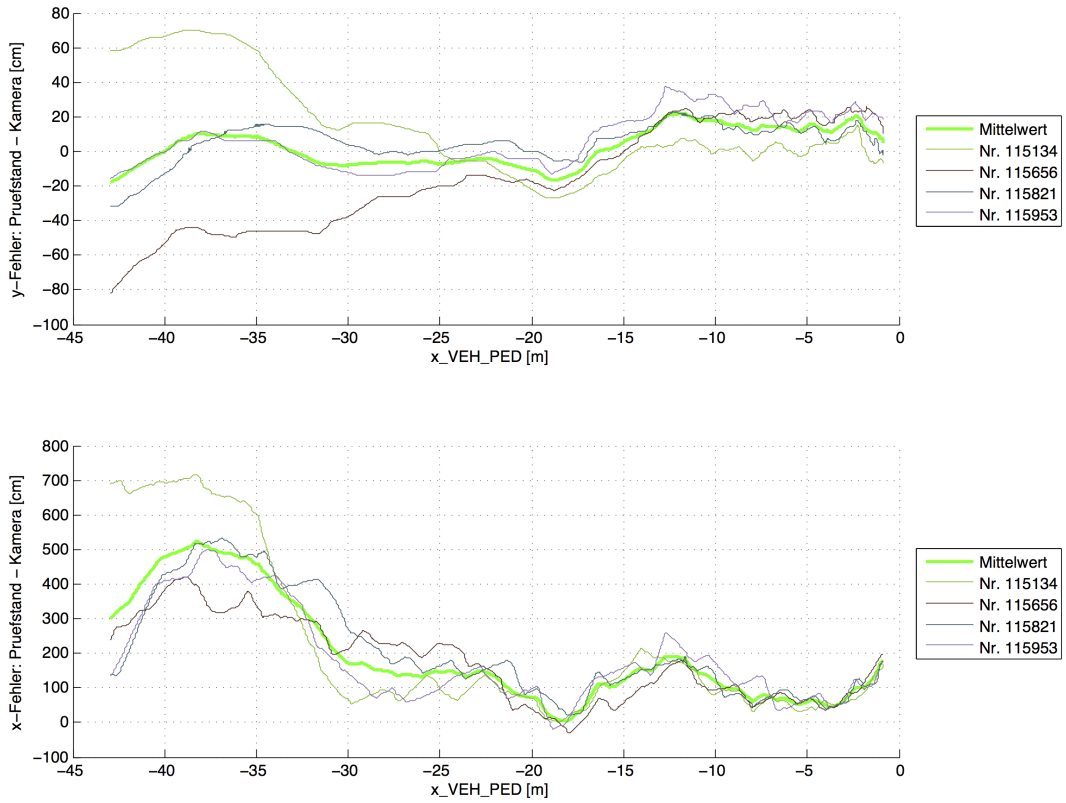

Abbildung A.13: y/x-Verlauf der Fehlerabweichung bei Ko-TAG

A.5 Szenario 5

Simulationsparameter

```
a_veh_start: -0.1 m/s^2   %Fahrzeug bewegt sich nicht mit konstanter Geschwindigkeit
v_veh_start: 28.6 km/h (t = -2 s) % Geschwindigkeit des Fahrzeuges 2 s vor Kollision
v_ped: 3.02 km/h % Fußgängergeschwindigkeit
y_ped_start: 1.49 m % Startposition des Fußgängers vom Fahrbahnmittelpunkt
x_veh_start: -18.83 m % Startposition des Fahrzeug-Nullpunktes vom Kollisionspunkt
x_veh_roadside: -3.33 m % x-Abstand des Verdeckungsfahrzeuges
y_veh_roadside: -30 m % y-Abstand des Verdeckungsfahrzeuges

Kamera:
Update-Rate Funktionsalgorithmus: 0.094 s
Bremszeitpunkt in der Simulation:  1.09 s   % Auslösung durch Ko-TAG bei der Messfahrt

Ko-TAG:
Update-Rate Funktionsalgorithmus: 0.05 s
```

Kamerabasierte Fußgängererkennung

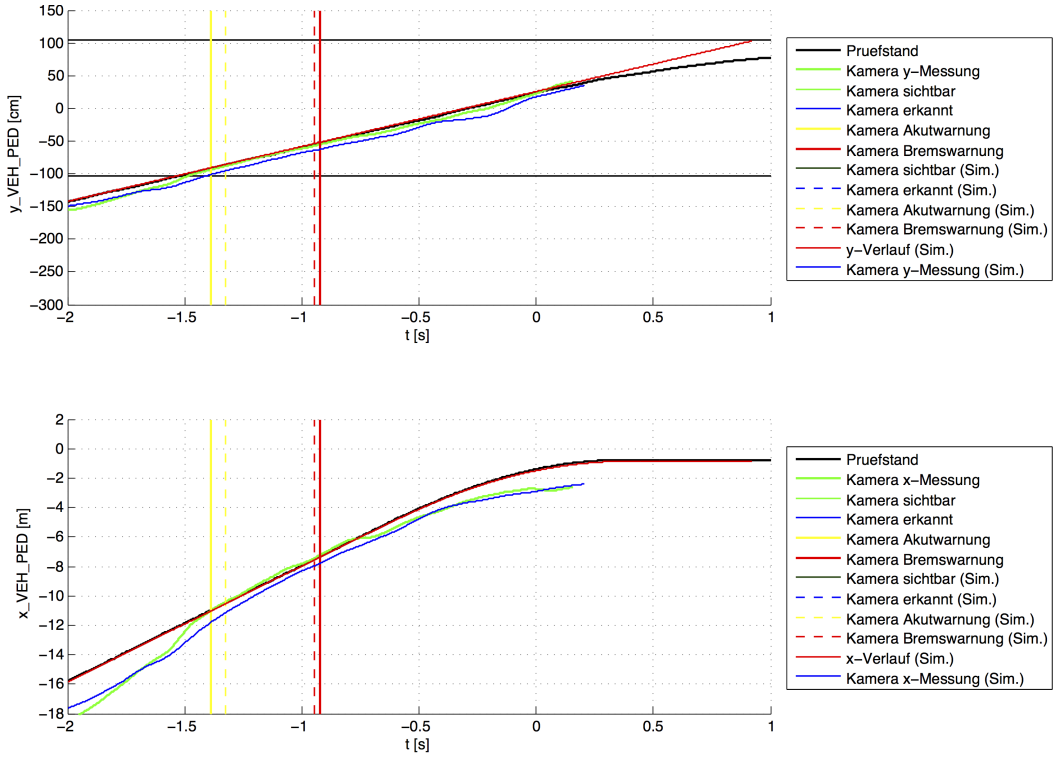

Abbildung A.14: y/x-Verlauf des Fußgängers mit der Kamera, Messung und Simulation

Nr.	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
120233	-1.38	-0.95	-11.05	-0.89	-0.54	-7.20
120556	-1.46	-0.97	-11.39	-0.96	-0.56	-7.45
120722	-1.38	-0.87	-10.98	-1.10	-0.65	-8.73
120858	-1.31	-0.85	-10.52	-0.74	-0.36	-6.00
MW	-1.39	-0.91	-10.98	-0.92	-0.53	-7.34
Sim.	-1.36	-0.88	-10.83	-0.89	-0.49	-7.14
Diff.	0.03	0.03	0.15	0.03	0.04	0.20

Tabelle A.7: Mess- und Simulationsergebnisse der Kamera

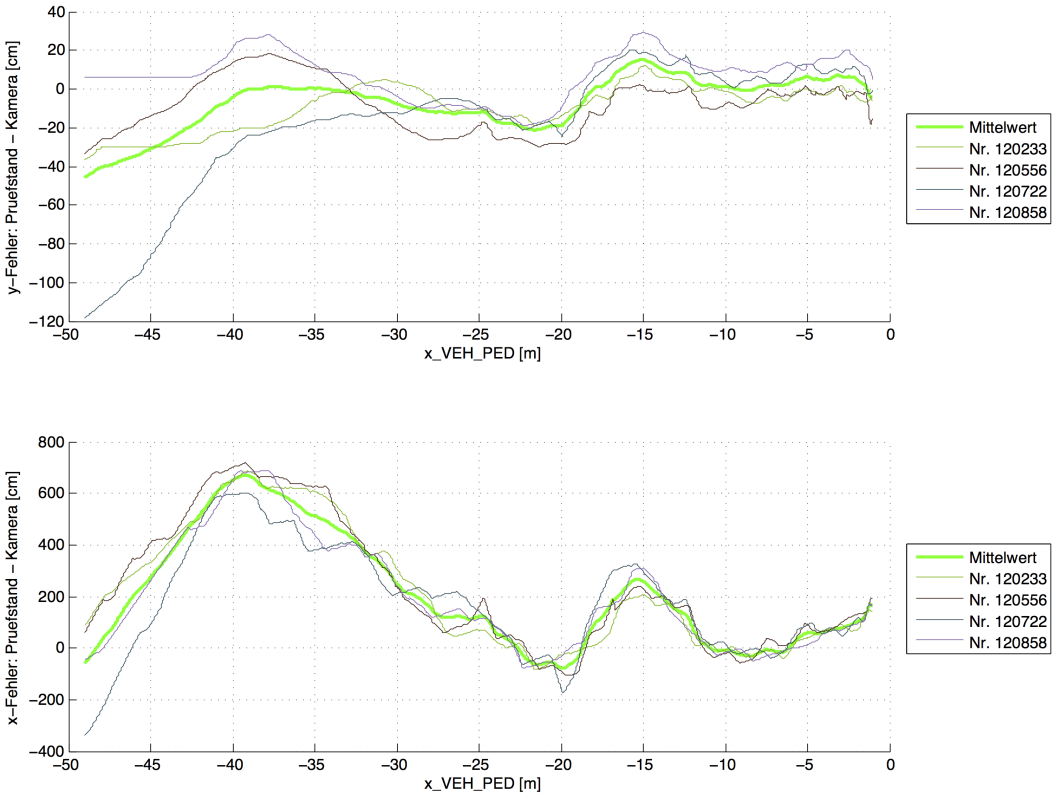

Abbildung A.15: y/x-Verlauf der Fehlerabweichung der Kamera

Ko-TAG-basierte Fußgängererkennung

Nr.	Akutw. [s]	y_PED [m]	x_VEH [m]	Bremsung [s]	y_PED [m]	x_VEH [m]
120233	-1.48	-1.03	-11.80	-1.06	-0.68	-8.52
120556	-1.49	-1.00	-11.63	-1.08	-0.66	-8.40
120722	-1.52	-1.01	-12.09	-1.04	-0.62	-8.35
120858	-1.51	-1.04	-12.09	-1.09	-0.69	-8.82
MW	-1.50	-1.01	-11.90	-1.07	-0.66	-8.52
Sim.	-1.46	-1.03	-11.99	-1.01	-0.64	-8.36
Diff.	0.04	-0.02	-0.09	0.06	0.02	0.16

Tabelle A.8: Mess- und Simulationsergebnisse des Ko-TAG-Sensors

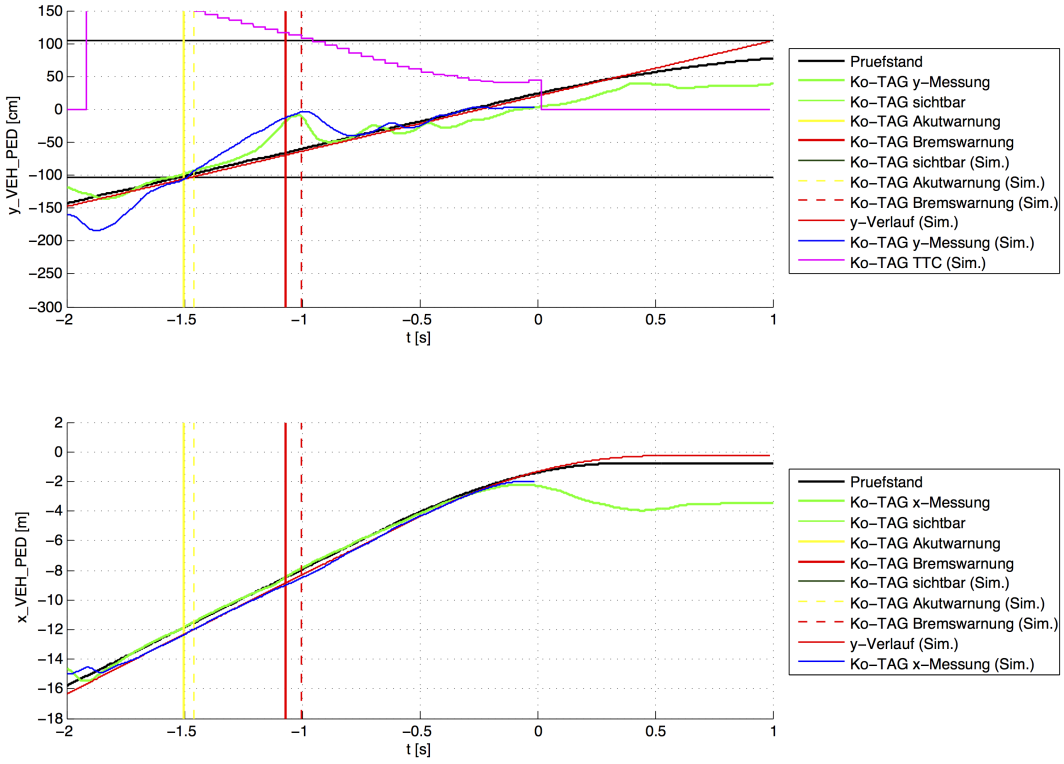

Abbildung A.16: y/x-Verlauf des Fußgängers mit Ko-TAG, Messung und Simulation

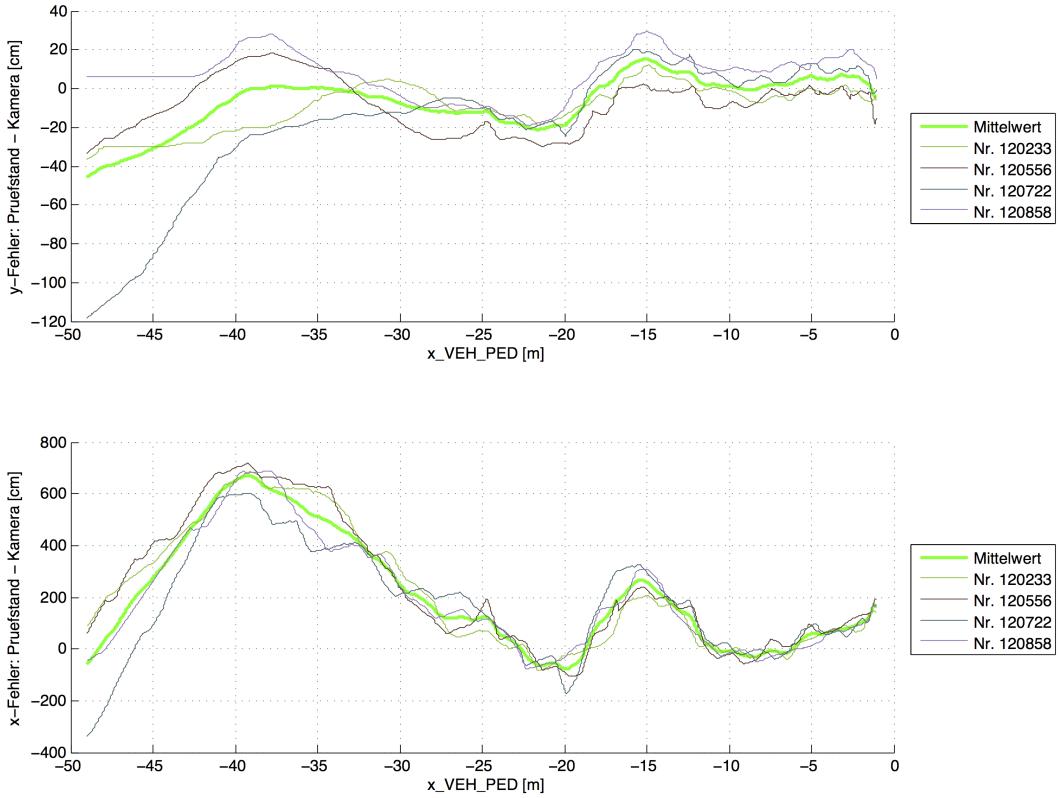

Abbildung A.17: y/x-Verlauf der Fehlerabweichung bei Ko-TAG