

Studiengang: Softwaretechnik
Prüfer: Prof. Dr. rer. nat. habil. Paul Levi
Betreuer: Dr. rer. nat. Dirk Reichardt
Beginn: 07. März 2001
Ende: 05. September 2001
CR-Klassifikation: G.3, I.2.9, I.2.10, I.2.11, I.6

Diplomarbeit-Nr. 1926

**„Entwurf von Fusionsobjekten
für den Einsatz in
Fahrerassistenzsystemen“**

André Schammer

Durchgeführt bei der DaimlerChrysler AG
Forschungsbereich FT3/AA, Esslingen

Universität Stuttgart
Fakultät Informatik

Danksagung

Zuerst bedanke ich mich bei Prof. Dr. rer. nat. habil. Paul Levi, der mir die Durchführung der Diplomarbeit in der Forschungsabteilung FT3/AA der DaimlerChrysler AG ermöglicht hat und mir während der Arbeit oft betreuend beiseite stand.

Besonders bedanken möchte ich mich auch bei meinem Betreuer Dr. Dirk Reichardt für seine guten Anregungen und Hinweise.

Ein weiterer Dank gilt Axel Gern und Steffen Görzig für ihre Ideen und Unterstützungen.

Ein großer Dank geht an Hans Breckle vom FZI in Karlsruhe für die gute Zusammenarbeit.

Bedanken möchte ich mich auch bei den Studenten und Mitarbeitern der Abteilungen FT3/AA und FT3/AB für das gute Arbeitsklima und die Gespräche.

Ich danke an dieser Stelle meinen Eltern und meiner Freundin, die mir zu der Zeit der Diplomarbeit beistanden. Gott danke ich für die Kraft, Geduld und Gesundheit, die er mir während der gesamten Zeit gab.

Inhaltsverzeichnis

1	Einführung	1
1.1	Motivation	1
1.2	Aufgabenstellung	3
1.3	Überblick über die Arbeit	4
2	Grundlagen	5
2.1	Fahrerassistenzsysteme	5
2.2	Multiagentensysteme	7
2.2.1	Merkmale eines Agenten	7
2.2.2	Merkmale eines Multiagentensystems	8
2.3	ANTS - eine generische Softwarearchitektur für Multiagentensysteme	9
3	Spezifikation einer Fusionsarchitektur für Fahrerassistenzsysteme	12
3.1	Studie „Konzeption und Spezifikation von Fusionsobjekten“	12
3.1.1	Grundlagen der Fusion von Sensordaten	13
3.1.2	Kontrollobjekte	16
3.1.3	Fusionsobjekte	17
3.1.4	Weltrepräsentation	17
3.2	Anwendung	22
3.2.1	Mögliche Sensorausstattung des Versuchsfahrzeugs	22
3.2.2	Akquirierung von Informationen durch die Sensorik	23
3.2.3	Möglichkeiten der Sensordatenfusion	29
3.3	Generische Klassenbibliothek für Fusionsvorgänge	30
3.3.1	Klassenhierarchie der Kontrollobjekte	30
3.3.2	Klassenhierarchie der Fusionsobjekte	33
3.3.3	Verwendung der Klassenbibliothek	33

4	Entwurf und Implementierung	38
4.1	Softwarearchitektur	39
4.2	Simulationsumgebung	41
4.2.1	Simulation	41
4.2.2	Erweiterungen der Simulation	42
4.2.3	Schnittstellen zur Simulation realer Sensoren	44
4.3	Fusionsagent	46
4.3.1	Zyklus des Fusionsagenten	46
4.3.2	Verwendung der Kontrollobjekte	47
4.3.3	Fusionsmethoden	49
4.4	Weltmodellagent	52
4.4.1	Zyklus des Weltmodellagenten	53
4.4.2	Datenstruktur des Weltmodells	54
4.4.3	Realisierung der Weiterführung und Prädiktion von Objekten	66
5	Test und Resultate	72
5.1	Testszenarien	72
5.2	Fusionsagent	74
5.3	Weltmodellagent	77
5.4	Gesamtsystem	81
6	Zusammenfassung und Ausblick	83
6.1	Zusammenfassung	83
6.2	Ausblick	84
6.2.1	Fusionsagent	84
6.2.2	Weltmodellagent	85
6.2.3	Gesamtsystem	85
A	Symbolische Postionsangaben	86
B	Koordinatensysteme, relative Abstände und Winkel	88

Kapitel 1

Einführung

1.1 Motivation

Durch den wachsenden Anspruch auf Komfort und Sicherheit in Straßenfahrzeugen sind die Automobilhersteller herausgefordert, Systeme zu entwickeln, die dem Fahrer durch Informationen und Eingriffe in die Fahrzeugsteuerung Hilfestellungen geben. Solche Fahrerassistenzsysteme müssen zuverlässig funktionieren, um dem Fahrer wirklich nützlich zu sein. Sie dürfen keinesfalls für den Fahrer oder andere Verkehrsteilnehmer eine zusätzliche Gefahr darstellen, sondern haben die Aufgabe, bestehende Gefahren im Straßenverkehr zu vermindern.

Zukünftige Fahrerassistenzsysteme sollen u. a. über das Wissen verfügen, welche Verkehrsteilnehmer sich im Umkreis des Fahrzeugs befinden oder wie die Fahrspur verläuft. Um dieses Umweltwissen zu erlangen, muss das Fahrzeug über Sensoren verfügen, mit denen dessen Umwelt beobachtet werden kann. Beispiele für solche Sensoren sind Kameras, Radargeräte und präzise digitale Straßenkarten.

In der Forschungsabteilung Fahrerassistenzsysteme FT3/AA der DaimlerChrysler AG werden neuartige Technologien für Fahrerassistenzsysteme entwickelt und getestet. Dafür stehen mehrere Versuchsfahrzeuge (in dieser Arbeit auch als Versuchsträger bezeichnet) zur Verfügung, die mit unterschiedlicher Sensorik ausgestattet sind. Durch Algorithmen, die auf den Sensordaten (z. B. auf Kamerabildern) angewendet wird, ist man bereits in der Lage, u. a. den Spurverlauf, andere Verkehrsteilnehmer (Fahrzeuge, Fußgänger, etc.), Verkehrszeichen und Ampeln zu erkennen. Die Konfiguration der für die Beobachtung der Umwelt benötigten Sensoren und Methoden ist vom Kontext (z. B. Autobahn oder Stadt), in dem sich der Versuchsträger befindet, und von der Aufgabe des Fahrerassistenzsystems abhängig.

Aktiv werden die Fahrerassistenzsysteme in den Versuchsträgern durch Bereitstellung von Informationen, Warnungen des Fahrers oder durch direkten Eingriff in die Fahrzeugregelung (Längs- und Querregelung). Dadurch ist das autonome Fahren unter bestimmten Bedingungen bereits möglich.

Der bisherige Systemaufbau eines Fahrerassistenzsystems wird in der Abbildung 1.1 skizziert. Auf den ermittelten Sensordaten werden direkt Verarbeitungsmethoden angewendet. Die Auswahl und Abfolge der Verarbeitungsmethoden ist fest programmiert. Dies hat den großen Nachteil, dass das System nicht selbständig auf geänderte Bedingungen (veränderter Kontext, neue Aufgaben) reagieren kann und das Metawissen über die angewendeten Problemlösungsprozesse dem System

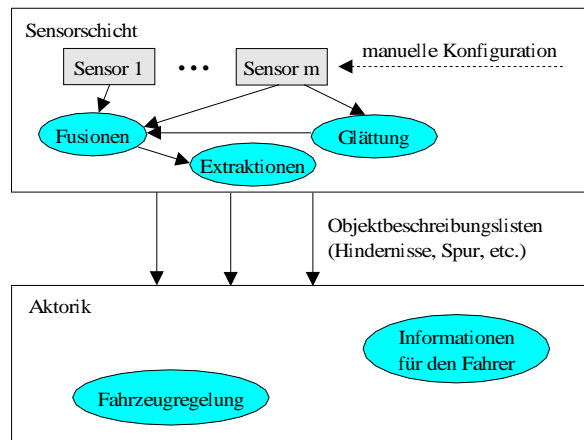


Abbildung 1.1: Bisherige Architektur eines Fahrerassistenzsystems

nicht bekannt ist. Ein einfaches Beispiel soll dieses Problem verdeutlichen. Ein Fahrzeug ist ausgestattet mit einer Kamera für die Sicht nach vorn. Die Aufgabe des Fahrerassistenzsystems soll sein, den Spurverlauf zu erkennen. Dafür stehen in diesem Beispiel zwei Methoden zur Spurerkennung zur Verfügung. Die Methode 1 liefert bei Sonnenschein optimale Ergebnisse, die Methode 2 wurde für schlechte Sicht bei Regenwetter entwickelt. Weiterhin wird davon ausgegangen, dass das aktuelle Wetter dem System bekannt ist. Der Wunsch ist, dass das System selbst die Methode auswählt, die unter den momentanen Witterungsbedingungen benötigt wird. In der bisherigen Systemarchitektur würde eine solche bedingte Methodenauswahl manuell durchgeführt oder fest programmiert werden. Bei komplexeren Zusammenhängen, wie sie bei der Multisensorfusion und bei der Existenz vieler bedingter Abläufe entstehen, entsteht auf diese Weise ein schlecht erweiterbares, starres System. Ein flexibles System könnte in dem obigen Beispiel bei Bedarf auch beide Methoden anwenden und die besten Daten aus beiden weiterverarbeiten.

Ein weiterer Nachteil der bisherigen Systemstruktur ist das direkte Aufsetzen der Aktorenschicht auf die Sensorausgaben. Das Weltmodell besteht aus einfachen Objektlisten, die durch die Sensordatenverarbeiter erstellt wurden. Die Algorithmik der Aktorenschicht wird direkt anhand der Kenntnis der zugrundeliegenden Aktorik realisiert. Ändert sich die Sensorenschicht z. B. durch den Ausfall eines Sensors, hat das direkten Einfluss auf die Funktionalität der Aktorenschicht. Wünschenswert ist eine Zwischenschicht, durch die ein einheitliches, konsistentes Umweltwissen hergestellt wird, das von den zugrundeliegenden Sensoren abstrahiert. Auf diesem Umweltwissen könnte die Aktorenschicht aufbauen, ohne die Informationen, durch welche Sensorik und Methoden das Wissen akquiriert wurde. Änderungen in der Sensordatenverarbeitung hätten keine direkten Auswirkungen mehr auf andere Komponenten des Systems.

Zur besseren Strukturierung eines Fahrerassistenzsystems und zur Realisierung der Flexibilität der einzelnen Systemkomponenten, damit sich das System selbständig zur Lösung neuer Aufgaben und an einen neuen Kontext anpassen kann, wurde in [Levi2000] eine Architektur für Fahrerassistenzsysteme vorgeschlagen, die sich an den Eigenschaften von Multiagentensystemen orientiert. Im Speziellen wurde in einer darauf aufbauenden Studie [Levi2001] die mögliche Struktur einer Fusionskomponente für die Sensordatenverarbeitung eingeführt, die Anpassungen an neue Bedin-

gungen zulässt und selbst vornimmt. Die Aufgaben, die in diesem Zusammenhang in dieser Arbeit zu bearbeiten waren, werden im folgenden Kapitel beschrieben.

1.2 Aufgabenstellung

Ziel der vorliegenden Arbeit ist die prototypische Realisierung eines Systems, das Sensordaten und Algorithmen flexibel fusionieren kann, die Fusionsresultate in Fusionsobjekten speichert und die Weltmodellierung als Grundlage der Situationsinterpretation durchführt (Abbildung 1.2). Als Ba-

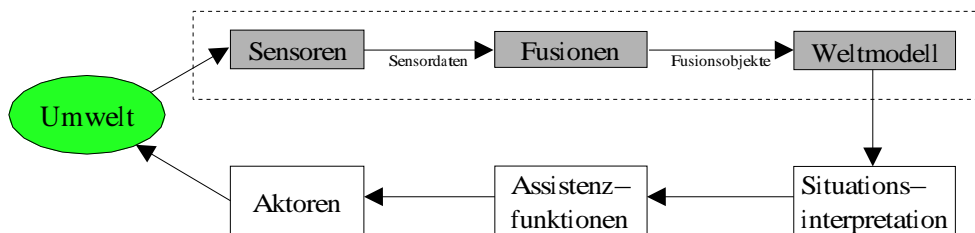


Abbildung 1.2: Grobe Architektur des zu entwickelnden Systems

sis dieser Arbeit dient die Studie “Konzeption und Spezifikation von Fusionsobjekten als Schnittstelle zwischen der Fusionsebene und der Situationsinterpretation“ [Levi 2001]. Den Kontext bildet ein autonomes Fahrzeug, das mit Sensoren für die Erfassung der Umwelt- und Eigendaten ausgestattet ist.

Aus der Aufgabenstellung bilden sich folgende Teilaufgaben:

- Untersuchung, wie die durch die Studie [Levi 2001] gegebene theoretische Grundlage bzgl. Sensornetzwerk, Sensordaten, Fusionsmethoden und Weltmodell in der Praxis angewendet werden kann
- Entwurf des Systems für die Implementierung unter Berücksichtigung einer Multiagentenarchitektur (ANTS) und der Offenheit für spätere Erweiterungen
- Erweiterung einer Simulation als Sensordatenquelle und deren Einbettung in die Multiagentenarchitektur zur realitätsnahen Erprobung der Fusions- und Weltmodellkonzepte
- Testimplementierung einer Fusionskomponente unter Verwendung einer gegebenen flexiblen Fusionsarchitektur
- Entwurf und Implementierung eines Weltmodells mit folgenden Komponenten:
 - Repräsentation und Verwaltung der Fusionsobjekte in einem topologischen Netz
 - Entwurf der Datenstrukturen für die Speicherung des Kontextes (Straße, Wetter, Lichtbedingungen, etc.) und der eigenen Daten (Fahrzeugdaten, Fahrtabsichten, Transaktionen, etc.)
 - Weiterführung von Objekten der Umwelt, die sich in “toten Winkeln“ befinden
 - Prädiktion der Position und des Verhaltens einzelner Objekte

- Auskunft über das Wissen über Gebiete in der Umwelt

Die Aufgaben werden durch wenige Beispielszenarien eingegrenzt. Eine sehr wichtige Anforderung ist die Offenheit des zu entwickelnden Systems, die spätere Erweiterungen und Änderungen zulässt.

Durch Tests des Gesamtsystems anhand weniger zu entwickelnden Szenarien soll vor allem die Flexibilität der Fusionsystemkomponente und die Konsistenz des Weltmodells untersucht werden.

Die zu erstellenden Systemkomponenten sollen ohne großen Aufwand in ein reales Versuchsfahrzeug portiert werden können.

1.3 Überblick über die Arbeit

Im Kapitel 2 wird zu Beginn eine kurze Einführung in Fahrerassistenzsysteme gegeben. Danach werden Konzepte von Multiagentensystemen und ANTS, einer Softwarearchitektur für Assistenzsysteme, beleuchtet.

Das nächste Kapitel geht auf die Grundlagen für das in dieser Arbeit entwickelte System ein. Dabei werden zuerst die in der Studie [Levi2001] gegebenen theoretischen Grundlagen der Sensordatenfusion, das Konzept für eine flexible Fusionsarchitektur und eine Methodik zur Weltmodellierung beschrieben. Danach wird die mögliche Sensorausstattung eines Versuchsträgers vorgestellt, anhand der die Grundlagen der Studie umgesetzt werden könnten. Dem folgt eine Auflistung von Merkmalen zur Beschreibung der Umwelt und des Versuchsträgers selbst. Dabei ist angegeben, welcher Sensor prinzipiell Informationen zu den Merkmalen beitragen kann. Am Ende des Kapitels wird die Struktur und die Handhabung einer Klassenbibliothek zur flexiblen Konfiguration von Fusionsvorgängen erklärt.

Die Realisierung des Systems wird im Kapitel 4 erläutert. Zu Beginn wird die entworfene Architektur vorgestellt, die für Fahrerassistenzsysteme neuartig ist. Danach wird u. a. auf die Realisierung des Fusionsagenten, der die Fusion der Sensordaten zu einzelnen Fusionsobjekten vornimmt, und den Weltmodellagenten eingegangen.

Anschließend wird im Kapitel 5 der Test des entwickelten Systems behandelt. Dabei werden speziell die Umsetzung der Ideen der Studie [Levi2001] und deren Bewertung betrachtet.

Abschließend wird im Kapitel 6 eine Zusammenfassung und ein Ausblick für weiterführende Arbeiten gegeben.

Kapitel 2

Grundlagen

In diesem Kapitel werden einige Grundlagen vorgestellt, die das Verständnis der in dieser Arbeit behandelten Thematik erleichtern und deren Kontext beleuchten sollen. Zuerst wird allgemein auf Fahrerassistenzsysteme eingegangen. Danach wird das grundlegende Konzept von Multiagentensystemen beschrieben. Zuletzt wird ANTS, eine Software-Architektur für Multiagentensysteme, kurz beleuchtet.

2.1 Fahrerassistenzsysteme

Das Steuern von Fahrzeugen auf der Straße ist in der heutigen Zeit hauptsächlich die Aufgabe des Menschen. Der Fahrer ist selbständig verantwortlich für die Quer- und Längsregelung des Fahrzeugs unter Berücksichtigung des aktuellen Straßenverlaufs, der Route zum Ziel, der anderen Verkehrsteilnehmer, der Dynamik und Bedienung des zu steuernden Fahrzeugs, der Verkehrsregeln und anderer Bedingungen. Oft ist der Mensch überfordert, die Aufgabe des fehlerfreien Steuerns eines Fahrzeuges zu erfüllen, besonders in kritischen Verkehrssituationen. Dies bestätigen die hohen Unfallzahlen. Für das Jahr 1999 gab die Bundesanstalt für Straßenwesen (BASt) bekannt, dass allein auf deutschen Straßen etwa 2,5 Mio. Unfälle registriert wurden. Dabei wurden bei jedem sechsten Unfall Menschen verletzt oder getötet. Die Anzahl der durch Verkehrsunfälle getöteten Menschen betrug 1999 rund 7800 (was der Einwohnerzahl einer Kleinstadt entspricht), die der Verletzten rund 500000. Demnach sind 1999 durchschnittlich jeden Tag rund 21 Menschen im Straßenverkehr gestorben und 1370 verletzt worden. Hinzu kommen Sachschäden in Milliardenhöhe.

Ursachen für schwere Unfälle sind laut [May2000] u. a. die Unaufmerksamkeit des Fahrers, z. B. durch Einschlafen oder Ablenkung hervorgerufen, die Fehleinschätzung des Verhaltens anderer Fahrzeuge oder der Dynamik des eigenen Fahrzeugs und das Eintreten unvorhergesehener Ereignisse, wie z. B. Fußgänger oder Tiere auf der Fahrbahn. Das Fehlverhalten des Fahrers kann auch durch Alkohol, Unerfahrenheit, Leichtsinn oder Krankheit ausgelöst werden.

Durch Fahrerassistenzsysteme wird versucht, dem Menschen bei der Steuerung eines Fahrzeugs Aufgaben abzunehmen bzw. ihn dabei zu unterstützen. Einige der dadurch entstehenden Vorteile sind:

- Mehr **Sicherheit** beim Fahren und die damit verbundene Senkung der Unfallzahlen

- Erhöhter **Komfort** durch Verringerung der vom Fahrer durchzuführenden Tätigkeiten
- Steigerung der **Effizienz** des Verkehrs

Die Sicherheit im Fahrzeug ist, wie bereits durch die oben aufgeführten Unfallszahlen motiviert, ein sehr bedeutender Punkt. Ihr galten bereits viele technische Entwicklungen, wie z. B. der Airbag oder das Antiblockiersystem (ABS). Ein Beispiel für ein Fahrerassistenzsystem, durch das die Sicherheit erhöht werden soll, ist der Spurassistent (Lane Departure Warning) für Lastkraftwagen. Durch eine Kamera wird die Position des Fahrzeugs innerhalb der Spur ermittelt. Anhand der Position in der Spur, der Fahrzeuggeschwindigkeit und dem Blinkerstatus wird festgestellt, ob die Gefahr besteht, dass der LKW ungewollt die Spur verlässt. Droht das ungewollte Überfahren einer Fahrbahnmarkierung, wird der Fahrer durch ein markantes Geräusch, das von der jeweiligen Gefahrenseite gesendet wird, gewarnt.

Erhöhter Komfort wird dem Fahrer beispielsweise durch den Abstandsregelautomaten Distronic geboten. Dabei kann der Abstand zu einem vorausfahrenden Fahrzeug automatisch konstant gehalten werden, indem das eigene Fahrzeug selbständig die Längsregelung übernimmt.

Mit der Steigerung der Effizienz sind z. B. die Senkung der Fahrzeiten und Kraftstoffkosten durch den Einsatz von Fahrerassistenzsystemen gemeint. Ein Beispiel dafür ist die automatische Routenführung mit Hilfe eines Navigationssystems, wenn es um die Umgehung eines Staus geht.

Die Art und Weise, wie ein Fahrerassistenzsystem agiert, ist unterschiedlich. Es kann durch die Lieferung nützlicher Informationen, durch eindeutige Warnungen bei kritischen Situationen oder durch Eingriffe in die Steuerung des Fahrzeugs assistierend aktiv werden. Dabei liegen dem System Informationen über die Wünsche und Vorlieben des Fahrers vor. Erst wenn das Fahrerassistenzsystem sich so verhält, wie der Fahrer es sich wünscht und erwartet, kann es ihm zu einer echten Hilfe werden, die er gern annimmt.

Im Folgenden wird ein Beispiel für ein mögliches Fahrerassistenzsystem vorgestellt, bei dem ein komplexeres Umweltwissen nötig ist, um sicher agieren zu können. Es handelt sich um einen Spurwechselassistenten, dessen Einsatzgebiet die Autobahn ist. Die Assistenzfunktion kann z. B. in der Information für den Fahrer liegen, ob ein Spurwechsel möglich ist, oder sogar in der automatischen Durchführung des Spurwechsels. Notwendig dafür ist das sichere Wissen des Assistenzsystems über die Verkehrssituation, den Spurverlauf, die geltenden Verkehrsregeln, die eigenen Fahrzeugdaten und das Fahrerprofil des Fahrers. Die Abbildung 2.1 zeigt eine Szene, in der ein solcher Spurwechselassistent Anwendung finden könnte. Zu sehen ist eine dreispurige

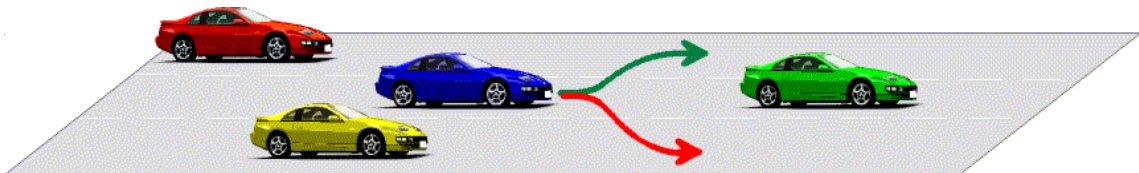


Abbildung 2.1: Spurwechsel auf einer Autobahn

Autobahn. Das Fahrzeug in der Bildmitte möchte durch einen Spurwechsel das Überholen des vorausfahrenden Fahrzeugs einleiten. Dazu stehen die linke und die rechte Spur zur Verfügung. Der Spurwechselassistent könnte z. B. grünes Licht zum Überholen über die linke Spur geben, da dort noch eine Lücke ist und die Verkehrsregeln nicht verletzt werden. Dagegen würde er

vermutlich den Wechsel auf die rechte Spur nicht empfehlen, da zum einen die Gefahr der Kollision mit dem rechtsfahrenden Fahrzeug besteht und das Überholen auf der rechten Seite ab einer bestimmten Geschwindigkeit nicht erlaubt ist.

Dazu muss während der Fahrt zu jeder Zeit die gültige Verkehrssituation bekannt sein. Für das Beobachten der Umwelt ist die Sensorik zuständig. Wissenslücken müssen durch Software gefüllt werden. Die zuverlässige Erkennung der Umwelt in jeder möglichen Fahrsituation stellt eine große Herausforderung für laufende und zukünftige Entwicklungen von Fahrerassistenzsystemen dar.

2.2 Multiagentensysteme

Eine Software-Architektur für komplexe Fahrerassistenzsysteme, in denen Sensordatenverarbeitung, eine Weltmodellierung und Handlungsableitungen vorgenommen werden, hat nach [Levi2000] besondere Anforderungen. Diese sind u. a. die Möglichkeit der parallelen, konkurrierenden Abarbeitung von Aufgaben, die Autodiagnose bei Fehlfunktionen oder Ausfällen, die Robustheit des Systems bei schlechten Sensordaten, die weitgehende Fehlerfreiheit der beteiligten Software und die Echtzeitfähigkeit. Durch eine Multi-Agenten-Architektur werden diese Anforderungen erfüllt. Das in dieser Arbeit entwickelte System ist in seinem Aufbau an das Paradigma eines Agentensystems angelehnt. Aus diesem Grund wird in diesem Abschnitt näher auf Multiagentensysteme eingegangen. Zuerst wird untersucht, was in diesem Zusammenhang unter einem Agenten zu verstehen ist. Danach wird auf grundlegende Prinzipien eines Systems eingegangen, das aus mehreren Agenten besteht und deshalb Multiagentensystem genannt wird.

2.2.1 Merkmale eines Agenten

Ein Agent ist eine autonome Einheit, die innerhalb eines Systems eine oder mehrere Aufgaben durchzuführen hat. In [Levi2000] werden einige Eigenschaften eines Agenten aufgezählt, wobei nicht jeder Agent alle dieser Eigenschaften besitzen muss:

Autonomie

Ein Agent besitzt die Kontrolle über seine Aktionen während seiner Ausführung und kann die Gestaltung des Problemlösungsprozesses selbst beeinflussen.

Interaktion

Agenten kooperieren mit anderen Agenten, um z. B. Daten auszutauschen oder gemeinsam Entscheidungen zu treffen.

Nebenläufigkeit und Asynchronität

Die Aktivitäten der Agenten müssen nicht sequenziell gekoppelt sein, d. h. die Agenten können zeitlich parallel agieren. Nachrichten an andere Agenten können ohne Wartezeit abgesetzt werden, es sei denn, eine schnelle Antwort ist für den korrekten Ablauf des Systems unabdingbar.

Lokalität

Jeder Agent enthält ein eigenes Weltmodell. Dieses entspricht dem Wissen, das er für seinen

Problemlösungsprozess benötigt. Wissen, das global für alle Agenten verfügbar sein soll, wird durch einen dafür verantwortlichen Agenten verwaltet und zur Verfügung gestellt.

Delegation

Agenten können Teilprobleme an andere Agenten weiterleiten, die für die Art und Weise der Abarbeitung selbst verantwortlich sind.

Reflektivität

Ein Agent muss seinen Zustand und den inneren Ablauf des Problemlösungsprozesses reflektieren können.

Strukturelle Offenheit

Ein Agent kann während seiner Ausführung neues Verhalten zeigen, indem er seinen Problemlösungsprozess modifiziert oder die Beziehungen zu anderen Agenten reorganisiert.

Die internen Abläufe und Datenstrukturen eines Agenten sind gekapselt und damit nach außen nicht sichtbar. Diese Lokalität unterstützt die Anforderung der Nebenläufigkeit. Verschiedene Algorithmen können gleichzeitig auf verschiedenen Datenstrukturen ablaufen, wodurch die Leistung des System gesteigert wird. Die Konsistenz der global verfügbaren Daten muss durch einen separaten Agenten gewährleistet werden, der die Weltmodelle der verschiedenen Agenten abgleicht. Die Autonomie, Reflektivität und strukturelle Offenheit sind wichtige Anforderungen an einen Agenten, der sich neuen Aufgabenstellungen oder veränderten Umweltbedingungen selbständig anzupassen hat. Dazu ist Metawissen über den Ablauf des Problemlösungsprozess notwendig. Durch die strukturelle Offenheit ist es dem Agenten möglich, sein Verhalten an bisherige Erfahrungen anzupassen, was auch durch die Fähigkeit, "lernen" zu können, ausgedrückt wird. Die Interaktion mit anderen Agenten oder dem Benutzer des Systems ist für einen Agenten wichtig, um Informationen und Aufträge erhalten und empfangen zu können.

2.2.2 Merkmale eines Multiagentensystems

Ein Multiagentensystem ist ein System, das aus mehreren interagierenden Agenten besteht. Durch die Lokalität und Autonomie der Agenten ist die Intelligenz des Systems auf einzelne Komponenten verteilt. Die Aufgabenzerlegung in Teilaufgaben und die Zuteilung dieser zu verschiedenen Agenten ist ein wichtiger Aspekt bei Multiagentensystemen. Die Aufgabenzerlegung und Aufgabenzuteilung können sowohl durch den Programmierer beim Entwurf des Systems, als auch zur Laufzeit durch das System selbst erfolgen. Während der Aufgabendurchführung kann, je nach Abhängigkeit der Agenten/Teilaufgaben untereinander, ein unterschiedlich hoher Kommunikationsaufwand zwischen den Agenten vorkommen. Das Resultat der Gesamtaufgabe wird durch die Synthese der einzelnen Teilresultate gebildet.

In [Levi2000] werden Merkmale genannt, nach denen sich Multiagentensysteme klassifizieren lassen. Einige dieser Merkmale werden hier kurz wiedergegeben:

Aufgaben

Liegen dem System globale Probleme vor, die in Teilprobleme zu zerlegen sind oder sind die Probleme bereits individueller Art?

Anzahl der Agenten

Multiagentensysteme unterscheiden sich in der Anzahl der Agenten, die beim Problemlösungsprozess mitwirken. Die Anzahl der Agenten kann während der Lösung einer Aufgabe konstant sein oder variieren.

Granularität

Einzelne Agenten können einfache oder auch komplexe Aufgaben lösen. Die Granularität gibt an, welche "Größe" bzgl. der Fähigkeiten und Verhaltensweisen die einzelnen Agenten haben.

Heterogenität

Der Unterschied der einzelnen Agenten eines Systems in ihren Fähigkeiten und Verhaltensweisen wird mit der Heterogenität ausgedrückt. Sind die Agenten eines Systems gleichartig, ist das System homogen.

Organisatorische Verteilung

Agenten können unabhängige Individuen sein oder z. B. in Teams oder hierarchischen Gruppen organisiert werden.

Kooperationsbereitschaft

Agenten können miteinander verhandeln oder auch konkurrieren, z. B. um eine Ressource oder bei Entscheidungen.

Kopplung

Lösen mehrere Agenten gemeinsam ein Problem oder sind sie weitgehend unabhängig voneinander?

Für die optimale Konfiguration eines Multiagentensystems gibt es sicher keine allgemeinen Lösungen. Sie ist abhängig von den Zielen, die mit dem System verfolgt werden und den Aufgaben zum Erreichen der Ziele. Weitere zu berücksichtigende Bedingungen sind z. B. die Echtzeitanforderung, die verfügbaren Ressourcen (z. B. Anzahl der Rechner) und der Kommunikationsaufwand zwischen den Agenten. Diese Bedingungen können sich während der Laufzeit des Systems ändern. Deshalb sollte die Fähigkeit der Reorganisation des Systems und der einzelnen Agenten gegeben sein.

Im nächsten Abschnitt wird ANTS, eine Systemarchitektur für Multiagentensysteme, vorgestellt.

2.3 ANTS - eine generische Softwarearchitektur für Multiagentensysteme

Das in dieser Arbeit realisierte System wurde unter Verwendung von ANTS [Görzig98] entwickelt. ANTS steht für "Agent NeTwork System" und ist eine Softwareumgebung, die Komponenten bereitstellt, mit denen Agentensysteme realisiert werden können. Das Netzwerk wird durch mehrere Agenten in einem Multi-Agenten-System gebildet, die zum Erreichen eines Zieles zusammenarbeiten. Die Generik von ANTS ist durch die anwendungsunabhängige Struktur gege-

ben. ANTS kann z. B. sowohl die Softwarearchitektur für ein Fahrerassistenzsystem als auch für eine agentenbasierte Fußballsimulation sein.

ANTS bietet die Möglichkeit, ein Multiagentensystem als verteiltes System zu konzipieren und auszuführen. Die beteiligten Prozesse können auf mehrere Rechner verteilt werden. Zusätzlich steht eine verteilte Datenbank zum indirekten Austausch von Daten zwischen den Prozessen zur Verfügung. Darüberhinaus wird die direkte Kommunikation zwischen den Prozessen ermöglicht. Die Abbildung 2.2 zeigt die Hauptkomponenten von ANTS und deren Beziehungen. Über die

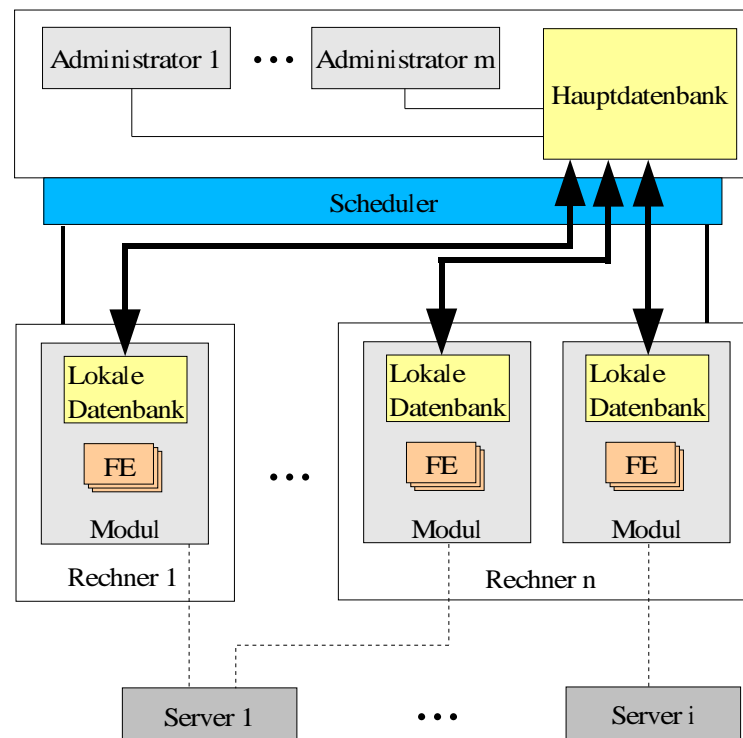


Abbildung 2.2: Komponenten von ANTS (FE = Funktionseinheit)

Hauptdatenbank werden die Daten, die auf die **lokalen Datenbanken** verteilt sind, abgeglichen. Dabei enthält die Hauptdatenbank Kopien aller Daten, die in den lokalen Datenbanken existieren. Eine lokale Datenbank verfügt nur über die Informationen, die für das Modul, dem sie zugeordnet ist, relevant sind.

Die Basisfunktionalität eines Multiagentensystems wird durch die **Funktionseinheiten** bereitgestellt. Sie verfügen über die Algorithmik, die zur Lösung von Teilaufgaben des Systems beiträgt. Z. B. kann eine Funktionseinheit die Aufgabe der Beschaffung von Sensordaten haben oder Merkmalsextraktionen auf Kamerabildern durchführen. Die Funktionseinheiten werden durch **Module** zusammengefasst. Dabei greifen Funktionseinheiten eines Moduls auf dieselbe lokale Datenbank zu. Neben der Bereitstellung der lokalen Datenbank übernimmt ein Modul die Laufzeitkontrolle für die ihm zugeordneten Funktionseinheiten und die Kommunikation zu anderen Komponenten des Systems. Die Module können über beliebig viele Rechner verteilt werden.

Die **Administratoren** haben direkten lesenden und schreibenden Zugriff auf die Hauptdatenbank. Sie verwalten die Ansteuerung der Funktionsmodule. Die Abarbeitung der Module und deren Funktionseinheiten kann dabei gestartet, gestoppt oder umkonfiguriert werden. Die dadurch mögliche flexible, dynamisch änderbare Konfiguration des Systems ist eine essentielle Anforderung an eine Software-Architektur für Multiagentensysteme. Die von den Administratoren getroffenen Entscheidungen über die Konfigurationen der Module werden dem **Scheduler** mitgeteilt, der für deren Umsetzung auf den einzelnen Rechnern zuständig ist. Darüberhinaus bietet er Informationen über die Zustände der Module auf den einzelnen Rechnern an.

Die **Server** sind spezielle Funktionseinheiten, auf die direkter Zugriff besteht. Sowohl ein Administrator als auch eine Funktionseinheit kann die Dienste eines Servers in Anspruch nehmen. Die Referenz auf einen Server ist in der Hauptdatenbank zu finden. Server sind eigenständige Programme, die von ANTS gestartet werden. Ein Beispiel für einen Server ist ein Prozess, von dem alle Komponenten des Systems jederzeit die aktuelle Zeit erfahren können (Zeitserver).

Die Initialkonfiguration des Systems wird durch eine **Skript-Sprache** beschrieben. In ihr kann festgelegt werden, welche Administratoren, Funktionseinheiten und Server auf welchen Rechnern gestartet werden sollen. Sie ermöglicht das Erstellen von Datenbankobjekten und das Zuweisen von Werten zu den Systemkomponenten und erstellten Objekten zur Startzeit. Nach der Initialisierung des Systems übernehmen die Administratoren die Konfiguration und Steuerung des Systems. Zusammenfassend werden einige Vorteile aufgelistet, die durch die Verwendung von ANTS entstehen:

- Die Konzeption eines Systems, in dem die Logik über mehrere Prozesse verteilt sein soll, wird erleichtert
- ANTS übernimmt die Parallelisierung der Prozesse
- Es steht eine verteilte Datenbank zur Verfügung, die selbständig die Kommunikation zwischen den Datenbankkomponenten übernimmt
- Es besteht die Möglichkeit, logische Abläufe während der Laufzeit zu verändern (dynamische Konfiguration durch die Administratoren)
- Von der physikalischen Umsetzung der Rechnerverteilung und Kommunikation kann abstrahiert werden
- Durch die Skript-Sprache kann ohne eine Neuübersetzung des Systems die Initialkonfiguration und die Verteilung des Systems über mehrere Rechner vorgegeben werden (statische Konfiguration)
- Durch die Plattformunabhängigkeit von ANTS kann das System einfach portiert werden

Ein Agent, wie er im Abschnitt 2.2.1 beschrieben wurde, kann bei der Verwendung von ANTS z. B. durch die Kombination eines Administrators mit einem oder mehreren Modulen realisiert werden. In den Funktionseinheiten ist seine funktionale Algorithmik implementiert, in den Administratoren die kontrollierende Logik über seinen Problemlösungsprozess.

Kapitel 3

Spezifikation einer Fusionsarchitektur für Fahrerassistenzsysteme

Im Rahmen der Neustrukturierung von Fahrerassistenzsystemen [Levi2000] wurden in [Levi2001] Konzepte zur Strukturierung der Fusions- und Weltmodellagenten vorgeschlagen. Diese Konzepte bilden die Basis für die im Kapitel 4 beschriebene prototypische Implementierung und werden im Abschnitt 3.1 vorgestellt, wobei auf die Grundlagen der Sensordatenfusion, die Strukturierung der Fusionsvorgänge und die Weltmodellierung eingegangen wird. Danach wird im Abschnitt 3.2 ein Beispiel für die Sensorik eines Fahrzeugs, auf dem ein Fahrerassistenzsystem zum Einsatz kommen könnte, gegeben. Die durch die Sensorik prinzipiell ermittelbaren Informationen über die Umwelt und die Eigendaten des Fahrzeugs werden nachfolgend beschrieben. Dies stellt gleichzeitig eine Erweiterung der in der Studie [Levi2001] begonnenen Zusammenstellung, welche Daten über die Umwelt und das eigene Fahrzeug für ein Fahrerassistenzsystem nützlich sein können, dar.

Anschließend wird im Abschnitt 3.3 eine in dieser Arbeit benutzte Klassenbibliothek vorgestellt, die Methoden zur Konfiguration von Fusionsvorgängen enthält.

3.1 Studie „Konzeption und Spezifikation von Fusionsobjekten“

In diesem Kapitel wird die Studie „Konzeption und Spezifikation von Fusionsobjekten“ [Levi2001] vorgestellt. Sie bildet die Hauptgrundlage für diese Arbeit. Ihr Ziel ist die systematische Beschreibung von Sensordatenfusionen unter Einführung von **Kontrollobjekten** für die Steuerung der Fusionsvorgänge und von **Fusionsobjekten**, in denen die Fusionsergebnisse abgelegt werden. Dadurch soll eine flexible und strukturierte Abfolge von Fusionen ermöglicht werden, die je nach Notwendigkeit der Akquirierung von Wissen und Verfügbarkeit der Sensoren angepasst werden kann. Weiterhin werden in der Studie die Einbettung der Fusionsobjekte in die Weltrepräsentation und verschiedene Ansätze zur Realisierung des Weltmodells betrachtet. Den Kontext bildet die agentenbasierte Softwarearchitektur eines autonomen Fahrzeugs. Die Verwaltung der Fusionsvorgänge wird dem Fusionsagenten zugeordnet, die Weltrepräsentation dem Weltmodellagenten.

Die Grundlagen für die Fusion von Sensordaten werden im Abschnitt 3.1.1 gegeben. Darauf

aufbauend wird im Abschnitt 3.1.2 die Spezifikation der Kontrollobjekte beschrieben. Danach folgt jeweils ein Abschnitt für die Fusionsobjekte (3.1.3) und für die Weltrepräsentation (3.1.4).

3.1.1 Grundlagen der Fusion von Sensordaten

Zum besseren Verständnis des Konzeptes der in der Studie [Levi2001] eingeführten Fusionsarchitektur werden in diesem Abschnitt die Grundlagen der Sensordatenfusion beschrieben. Ein Sensor ist in diesem Zusammenhang eine strukturierte Einheit, die aus einer Sensoreingabe (Messdatenerfassung), einer Weiterverarbeitung der Sensoreingabe und schließlich einer Sensorausgabe, die das Resultat der Verarbeitung enthält, besteht. Dementsprechend sind unter dem Begriff eines Sensors nicht nur *physikalische* Sensoren, sondern auch dazugehörige Algorithmen zur Verarbeitung der akquirierten Informationen zu verstehen. Sensoren, die auf den Ausgaben physikalischer Sensoren aufbauen, nennt man *logische* Sensoren.

In einem logischen Sensor, der Sensorausgaben mehrerer Sensoren verarbeitet und zu einer einheitlichen Darstellung verschmilzt, findet eine *Sensordatenfusion* statt. Um diese Sensordatenfusion geht es in den folgenden Abschnitten. Zuerst werden ihre Vorteile beleuchtet. Danach wird erläutert, in welche *Fusionsarten* man die Fusionen gliedern kann und wie ein *Sensornetzwerk* entsteht. Anschließend werden die für die hier vorgestellte Fusionsarchitektur sehr bedeutenden *Fusionsebenen* beschrieben.

3.1.1.1 Vorteile der Sensordatenfusion

Die Gewinnung von Informationen zur Erlangung von Umweltwissen zur Bearbeitung oder Lösung eines Problems ist nach [Jörg91] verbunden mit dem Wunsch nach Vollständigkeit, Genauigkeit, Sicherheit und Widerspruchsfreiheit. Benutzt man in einem System, das auf sensorischer Information basiert, mehrere Sensoren, können die genannten Eigenschaften des Umweltwissens eher erreicht werden, als wenn keine Sensordatenfusion stattfindet. Mehrere Sensoren können zur Vollständigkeit beitragen, wenn sie verschiedene Bereiche in der Umwelt abdecken und ihre Informationen sich dadurch ergänzen, oder aus einem überlappenden Bereich voneinander unabhängige Informationen ermitteln (*Komplementarität*). Weiterhin kann die Komplementarität z. B. auch Mehrdeutigkeiten auflösen. Eine weitere Verbesserung der sensorischen Informationsgewinnung wird durch die *Redundanz* von Sensoren, deren Sichtbereiche sich überlappen und die zur Gewinnung derselben Informationen herangezogen werden, bewirkt. Ebenso wird durch die Redundanz der Ausfall des Gesamt- oder eines Teilsystems durch den Wegfall eines einzelnen Sensors vermieden.

Ein weiterer Vorteil ist nach [Jörg91] und [Reichardt96] die durch Sensordatenfusion mögliche Leistungssteigerung des Systems, die für das Ziel der Realzeitfähigkeit essentiell ist.

Zusammenfassend sei nach [Levi2001] erwähnt, dass die Sensordatenfusion eine sehr wichtige Komponente dafür ist, ein System zuverlässig, verfügbar, robust, sicher und schnell zu machen.

3.1.1.2 Fusionsarten und Sensornetzwerk

Die verschiedenen Möglichkeiten, Sensordaten zu fusionieren, werden nach [Levi2001] Fusionsarten genannt. Es wird zwischen drei Fusionsarten unterschieden, die nachfolgend erklärt werden.

Komplementäre Fusion

Diese Art von Fusion findet statt, wenn die beteiligten Sensoren voneinander unabhängig sind. Diese Unabhängigkeit ist bei Sensoren, die das gleiche Phänomen messen (z. B. Abstand), gegeben, wenn sich ihre Akquisitionsbereiche nicht überlappen. Sensoren, die verschiedene Phänomene messen (z. B. Temperatur durch ein Thermometer und Helligkeit durch eine Kamera), sind, was die Messung betrifft, von vornherein voneinander unabhängig.

Konkurrierende Fusion

Wird die Akquisition der gleichen Umweltinformation im selben Bereich der Umwelt durch mehrere Messungen vorgenommen, besteht eine Konkurrenz zwischen den Messwerten, welcher dem realen Wert am nächsten kommt. Das Vereinheitlichen der verschiedenen Informationen zu einer, d. h. das Minimieren der Redundanz, wird konkurrierende Fusion genannt. Ein Beispiel dafür ist das Messen des Abstandes eines vorausfahrenden Fahrzeugs durch einen Radar- und einen Kamerasensor und der daraus resultierende Abstand, der z. B. eine Mittelung der Werte sein könnte.

Kooperative Fusion

Von kooperativer Fusion wird gesprochen, wenn ein benötigtes Resultat nicht nur von einem Sensor allein erbracht werden kann und erst die Kombination mehrerer Sensoren die Informationsgewinnung ermöglicht. Z. B. ist die Fusion zweier Monokularbilder zu einem Stereobild eine kooperative Fusion, da aus einem Stereobild mehr, bzw. genauere Information ermittelt werden kann (z. B. räumliche Ausmaße).

Die Fusionsarten sind nicht streng isoliert zu sehen. Eine Fusion kann eine Kombination der verschiedenen Fusionsarten darstellen. Eine solche *hybride* Fusion ist z. B. die komplementäre und zugleich kooperative Fusionsart, wenn zwei Kameras, deren Sehbereiche nebeneinander liegen, verwendet werden, um zusammen einen großen Sehbereich zu bilden.

Ein *Sensornetzwerk* entsteht dann, wenn man die einzelnen Sensoren als Knoten ansieht und die Verbindung zweier Knoten die verwendete Fusionsart bezeichnet. Ein Sensornetzwerk ist *dynamisch*, wenn sich die Anzahl der Sensoren und die Fusionsarten ändern können, sonst *statisch*. Werden nur identische Sensoren eingesetzt, ist das Sensornetzwerk *homogen*, bei verschiedenartigen Sensoren *heterogen*.

Ein weiterer Begriff ist die *Algorithmenfusion*. Diese findet statt, wenn auf dem gleichen Datensatz verschiedene Algorithmen Berechnungen durchführen und die Resultate der Algorithmen wieder zu einem Ergebnis fusioniert werden.

3.1.1.3 Fusionsebenen

Eine weitere Klassifizierung der Sensordatenfusion wird durch den Begriff der *Fusionsebene* vorgenommen. In hierarchisch aufsteigender Ordnung sind dies die *Signalebene*, die *Merkmalebene* und die *Entscheidungsebene*. Im Folgenden werden die Fusionsebenen näher beschrieben.

Signalebene

Fusionen auf der Signalebene werden bei Sensoren angewendet, die das gleiche physikalische

Phänomen in einem überlappenden Bereich messen. Ein Beispiel dafür ist die kooperative Fusion zweier Monokularbilder zu einem Stereobild. Weiterhin ist für die Signalebene die Extraktion von Merkmalen aus den Signalen (z. B. Abstand, Breite, etc.) typisch.

Merkmalsebene

Auf der Merkmalsebene werden Sensordaten fusioniert, die bereits vorverarbeitet wurden. Dadurch wird eine Vervollständigung der Umwelt- bzw. Objektbeschreibungen erreicht. Liefert z. B. der Radarsensor den Abstand und die Längsgeschwindigkeit eines Hindernisses in Form eines Merkmalvektors und der Stereokamerasensor die Ausmaße und die Farbe desselben, werden die verschiedenen Merkmale zu einem neuen, größeren Merkmalsvektor fusioniert (komplementäre Fusion). Auf der Merkmalsebene ist ebenso die konkurrierende Fusion möglich, indem z. B. die Abstandsangabe des Radarsensors mit der Abstandangabe der Stereokamera verglichen wird. Anhand des resultierenden Merkmalvektors können nun Hypothesen durch Identitätsbestimmungsalgorithmen gemacht werden, um welchen Typ von Objekt es sich handelt.

Entscheidungsebene

Auf der Entscheidungsebene werden Entscheidungen bzgl. der Identität von Objekten gefällt. Existieren beispielsweise mehrere Hypothesen über die Identität eines Objekts (z. B. PKW, LKW), wird in der Entscheidungsebene anhand geeigneter Algorithmen eine Entscheidung getroffen.

Die Fusionsebenen sind nicht isoliert zu betrachten. Das Bild 3.1 zeigt den Datenfluss zwischen den Ebenen. Die Eingabe der Signalebene sind die rohen Sensordaten. Diese werden

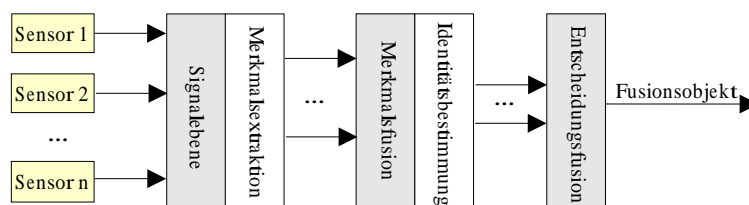


Abbildung 3.1: Fusionsebenen in einem Fusionsvorgang

fusioniert. Danach werden aus den Signalen Merkmale extrahiert und an die Merkmalsebene weitergereicht. Diese fusioniert die Merkmale und ordnet sie dabei gemeinsamen Objekten zu. Danach findet eine Identitätsbestimmung statt, unter evt. Zuhilfenahme mehrerer Algorithmen. Die daraus resultierenden Hypothesen werden schließlich in der Entscheidungsebene aufgenommen und es wird eine Entscheidung über die Objektidentität gefällt.

Folgende Kapitel geben einen Überblick über das in der Studie [Levi2001] vorgeschlagene Konzept, wie die Sensordatenfusion strukturiert, dynamisch änderbar und unter Berücksichtigung der Fusionsmerkmale (Fusionsebene, Fusionstechnik, etc.) in einem Fahrerassistenzsystem realisiert werden könnte.

3.1.2 Kontrollobjekte

Für die Steuerung bzw. Kontrolle der Fusionsvorgänge sind die Kontrollobjekte verantwortlich. Durch sie werden die Fusionsvorgänge während der Laufzeit konfiguriert und ausgeführt. Sie berücksichtigen für die Konfiguration der Fusionsvorgänge die im Abschnitt 3.1.1 genannten Merkmale einer Fusion, wie z. B. die Fusionsart und die Fusionsebene, das Sensornetzwerk, die Ziele, die Transaktionen und momentanen Steueraktionen des Systems, den Kontext (z. B. Wetter, Fahrbahnzustand, etc.) und alle verfügbaren Methoden. Diese Informationen und Referenzen werden in einer **Fusionsmerkmalsbibliothek** hinterlegt (Abbildung 3.2).

Teile der Fusionsmerkmalsbibliothek (z. B. Kontext, eigene Daten) sollen in der Realisierung des

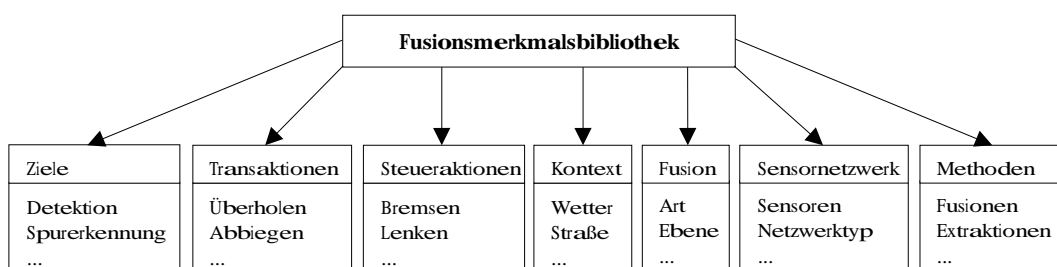


Abbildung 3.2: Ausschnitt aus der Fusionsmerkmalsbibliothek

Konzepts in der Weltrepräsentation gehalten werden, aber den Kontrollobjekten leicht zugänglich sein.

Weiterhin zeigen die Kontrollobjekte auf die bisherigen Fusionsresultate, die in der Studie als Fusionsobjekte bezeichnet werden (siehe dazu Abschnitt 3.1.3).

Es wird zwischen verschiedenen Arten von Kontrollobjekten unterschieden, denen jeweils ein Aufgabenbereich zugeordnet ist. Diese verschiedenen Arten werden nachfolgend kurz vorgestellt.

Prozesskontrollobjekt

Das Prozesskontrollobjekt bestimmt die Konfiguration der einzelnen Fusionsvorgänge. Es enthält einen Verweis auf die Fusionsmerkmalsbibliothek, kennt die Aufgaben, die durch die Fusionsvorgänge gelöst werden sollen und hat damit eine Wissensbasis für die durchzuführende Konfiguration. Neben der Konfiguration übernimmt es zusätzlich die Speicherung der Referenzen auf die Fusionsvorgänge, die durch Fusionskontrollobjekte beschrieben werden.

Fusionskontrollobjekt

Das Fusionskontrollobjekt ist für den Ablauf eines einzelnen Fusionsvorgangs verantwortlich. Ein Fusionsvorgang beinhaltet die Fusionen und Methodenkombinationen von der Signalebene bis hin zur Entscheidungsebene (siehe Abschnitt 3.1.1.3) in Form von Kontrollobjekten, die ihrer jeweiligen Ebene zugeordnet sind. Der gesamte Fusionsvorgang wird vom Prozesskontrollobjekt konfiguriert, das Fusionskontrollobjekt führt ihn aus. Weiterhin speichert ein Fusionskontrollobjekt die Referenzen auf die Zwischen- und Endergebnisse des Fusionsvorgangs.

Kontrollobjekt

Die Konfiguration einer Methode als Teil eines Fusionsvorganges wird allgemein durch ein Kontrollobjekt beschrieben. Ein Kontrollobjekt enthält Verweise auf eine auszuführende Methode und deren Eingabedaten. Für jede der Fusions Ebenen und für die Merkmalsextraktion aus Signalen und die Identitätsbestimmung aus Merkmalen werden spezialisierte Kontrollobjekte eingeführt.

Detaillierte Informationen über die Struktur und die Realisierung der Kontrollobjekte sind im Abschnitt 3.3 nachzulesen. Der folgende Abschnitt beschreibt die in der Studie vorgeschlagene Struktur und Aufgabe der Fusionsobjekte.

3.1.3 Fusionsobjekte

Für die Speicherung der Fusionsresultate einer Fusions Ebene sind die Fusionsobjekte zuständig. Sie vereinigen alle Ergebnisse der ausgeführten Methoden des Fusionskontrollobjekts einer Ebene, beziehen sich aber nicht auf eine darunter liegende Schicht. Für jedes erkannte Element in der Umwelt des Versuchsträgers (Spur, Hindernis, Verkehrszeichen, etc.) sollen dabei die Werte, die zu einem Element gehören, in einem separaten Fusionsobjekt zusammengefasst werden.

In der Studie werden die drei Klassen **Signalfusionsobjekt**, **Merkmalsfusionsobjekt** und **Entscheidungsfusionsobjekt** für die Fusionsobjekte konzipiert, die jeweils einer der Fusions Ebenen zugeordnet sind. Allen Klassen gemeinsam sind exemplarische Attribute für die kinematischen Werte (Position, Geschwindigkeit, etc.), die Prädiktion, die vermutete Absicht und das Fahrziel. Ebenso ist die Herkunft der im Fusionsobjekt hinterlegten Einzelmerkmale zu speichern, z. B. durch Angabe der Bildbereiche, in denen das Objekt erkannt wurde oder durch Verweise auf die Sensoren oder die Methoden, die im Fusionsvorgang mitgewirkt haben.

Ein Unterschied zwischen den Fusionsobjekten der verschiedenen Fusions Ebenen besteht in den Ergebnissen der Objektidentifikation. Während in einem Signalfusionsobjekt z. B. lediglich die Identifikation "Hindernis" eines erkannten Bildobjekts stehen kann, ist diese Information im Merkmalsfusionsobjekt bereits detaillierter (z. B. Rückansicht eines PKW) und im Entscheidungsfusionsobjekt wird eindeutig auf eine Objektkategorie geschlossen (z. B. PKW).

Die genaue Struktur der einzelnen Fusionsobjekte kann erst durch die jeweilige Anwendung, in der das in der Studie vorgestellte Konzept verwendet wird, festgelegt werden. Die Verwendung der Fusionsobjekte in dieser Arbeit ist in den Abschnitten 4.3 (Fusionsagent) und 4.4.2.4 (Objekte in der Umwelt) beschrieben.

Die Entscheidungsfusionsobjekte, welche die Resultate der kompletten Fusionsvorgänge beinhalten, bilden die Eingabe für die Weltrepräsentation, die im nächsten Abschnitt erläutert wird.

3.1.4 Weltrepräsentation

Unter den Begriffen "Welt" und "Umwelt" wird in dieser Arbeit die Umgebung eines Fahrzeugs verstanden. Die Umgebung beinhaltet u. a. die Straße, andere Fahrzeuge und sonstige Objekte (z. B. Fußgänger, Verkehrszeichen, Bäume, etc.), das Wetter, die Landschaft, die Lichtverhältnisse und die Absichten und Manöver von Objekten. Das Abbild dieser realen Welt, das im Fahrerassis-

tenzsystem gespeichert ist und nur einen Ausschnitt der realen Umwelt darstellt, wird in diesem Zusammenhang als Weltmodell bezeichnet. Zum Weltmodell gehören auch die eigenen Fahrzeugdaten.

Die Studie beschreibt einige grundlegende Methoden, mit denen die Weltmodellierung vorgenommen werden kann. Dabei steht die direkte Einbeziehung der Fusionsresultate in Form von Fusionsobjekten im Vordergrund, wobei davon ausgegangen wird, dass nur Objekte und nicht ihre Einzelmerkmale im Weltmodell gespeichert werden. Das bedeutet, dass z. B. beim Erkennen eines Blinkers auf ein Fahrzeug geschlossen wird und dieses im Weltmodell mit evt. Verweis auf den Blinker gehalten wird. Ein einzelner Blinker würde jedoch nicht zusammenhangslos dem Weltmodell übergeben werden, es sei denn, beim Blinker handelt sich um ein loses Teil auf der Straße, das ein Hindernis darstellt.

Die in der Studie beschriebenen Methoden basieren auf probabilistischen Netzen, durch die bedingte Wahrscheinlichkeiten und Zustände modelliert werden können. In diesem Abschnitt wird der Vorschlag einer topologischen Karte, die auf einem dynamischen probabilistischen Netz aufbaut, wiedergegeben. Diese topologische Karte wurde in dieser Arbeit zur Unterstützung der Prädiktion und Weiterführung implementiert (siehe Abschnitt 4.4.3). Zuerst wird kurz auf die theoretischen Grundlagen eingegangen. Darauf aufbauend wird die in der Studie [Levi2001] eingeführte topologische Karte betrachtet.

3.1.4.1 Markov-Ketten und probabilistische Netze

Eine *Markov-Kette* kann durch einen gerichteten Graphen mit gewichteten Kanten dargestellt werden. Die Knoten des Graphen entsprechen Zuständen. Die gewichteten Kanten beschreiben die mit Wahrscheinlichkeiten versehenen Zustandsübergänge. Ein Zustand S_t darf dabei nur das Resultat eines zeitlich unmittelbar davorliegenden Zustandes S_{t-1} sein, was durch die *Markov-Bedingung* besagt wird. Dieser Zusammenhang kann durch folgende Formel in der Schreibweise für bedingte Wahrscheinlichkeiten ausgedrückt werden:

$$P(S_t | S_{t-1}) \tag{3.1}$$

Die Abbildung 3.3 zeigt einen Beispielgraphen für eine Markov-Kette. Darin ist deutlich die erfüllte Markov-Bedingung zu sehen, da jeder Systemzustand nur von seinem zeitlichen Vorgängerzustand abhängt. Die Anwendung einer Markov-Kette soll folgendes Beispiel verdeut-

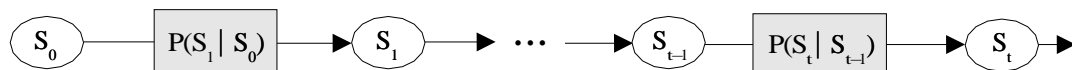


Abbildung 3.3: Darstellung einer Markov-Kette

lichen. Gegeben sind drei Zustände A, B und C. Ein System kann zu einem Zeitpunkt nur einen der Zustände annehmen. Der Systemzustand zum Zeitpunkt t wird durch S_t ausgedrückt. Für den Zustandswechsel wird eine Wahrscheinlichkeitsmatrix angenommen:

$$P = \begin{pmatrix} 0,5 & 0,2 & 0,3 \\ 0,3 & 0,4 & 0,3 \\ 0,4 & 0,2 & 0,4 \end{pmatrix} \tag{3.2}$$

Diese sagt aus, mit welcher Wahrscheinlichkeit das System, abhängig vom aktuellen Zustand, einen bestimmten Folgezustand einnimmt. Z. B. ist die Wahrscheinlichkeit für den Wechsel des Systems aus dem Zustand A in den Zustand B mit 0,2 definiert, wobei das Beibehalten des Zustandes A mit 0,5 angegeben wird. Damit die Markov-Kette kohärent ist, muss jede Zeilensumme gleich 1 sein. Es lässt sich nun folgender Zusammenhang erkennen:

$$S_t = S_{t-1}P = S_{t-2}P^2 = S_{t-n}P^n \tag{3.3}$$

Markov-Ketten bieten demnach eine Methodik an, wahrscheinlichkeitsbasierte Zustandsübergänge von Systemen zu modellieren und daraus ableitend Berechnungen durchzuführen.

Probabilistische Netze, auch als Bayes-Netze bezeichnet, repräsentieren Beziehungen zwischen Zuständen und Ereignissen. Qualitativ werden die Beziehungen in einem **DAG** (**D**irected **A**cyclical **G**raph)

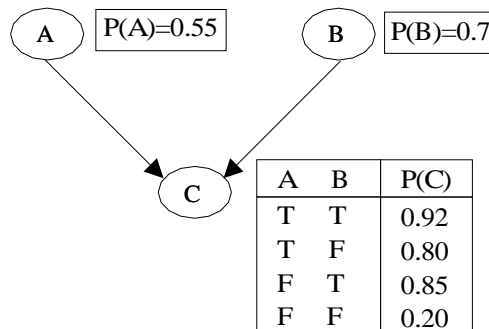


Abbildung 3.4: Probabilistisches Netz

Acyclical Graph) modelliert. Die Knoten entsprechen Zufallsvariablen. Die Kante eines Knotens P zu einem Knoten C gibt an, dass P den Knoten C beeinflussen kann. P ist damit ein Elternknoten von C. Die Stärke des Einflusses der Elternknoten auf einen Knoten wird quantitativ mittels einer **CPT** (**C**onditional **P**robability **T**able) beschrieben. Die Abbildung 3.4 zeigt ein einfaches probabilistisches Netz mit den zugehörigen CPTs. Die Knoten A, B und C modellieren 3 verschiedene Ereignisse. Das Ereignis C ist kausal von A und B abhängig. A und B sind voneinander unabhängig, deshalb ist zwischen ihnen keine Verbindung. Für die elementaren Ereignisse A und B sind jeweils die Wahrscheinlichkeiten für ihr Eintreten angegeben. Durch die CPT für das Ereignis C wird deutlich, wie A und B das Ereignis C beeinflussen. Umgangssprachlich formuliert bedeutet dies in dem Beispiel, dass die Wahrscheinlichkeit für das Eintreten des Ereignisses C höher ist, wenn A und/oder B eintreten, als wenn nur C eintritt.

Durch **Dynamische Bayes-Netze** (DBNs) ist es möglich, zeitliche Prozesse zu modellieren. Die Grundform eines DBN wird in der Abbildung 3.5 gezeigt. Jeder Zustand S_t ist demnach vom zeitlich vorhergehenden Zustand S_{t-1} abhängig. Dies erinnert an die oben vorgestellte Markov-Bedingung. Weiterhin kann man aus einem gegebenen Zustand weitere Informationen schlussfolgern, was in der hier dargestellten Grundform durch die Variable I dargestellt wird.

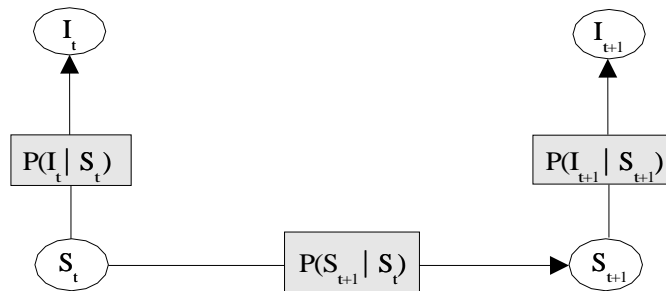


Abbildung 3.5: Grundform eines Dynamischen Bayes-Netz nach [Levi2001]

Die Berechnung eines neuen Zustandes aus seinem Vorgängerzustand heraus wird in diesem Zusammenhang als *Prädiktion* bezeichnet, die Ableitung der Information I aus einem Zustand S als *Schätzung*. Die Anwendung dieser Methodik für die Weltmodellierung wird im folgenden Abschnitt erläutert.

3.1.4.2 Topologische Karte

Für die Darstellung und Berechnung von Verkehrssituationen, die Aufgaben des Weltmodells sind, wurde in [Levi2001] eine Methodik vorgeschlagen, die auf probabilistischen Netzen beruht. Ein Zustand entspricht dabei der Menge der über ein Fahrzeug bekannten Merkmale (z. B. Geschwindigkeit, Ausmaße, etc.) zu einem Zeitpunkt. Die Abhängigkeit zwischen zwei aufeinanderfolgenden Zuständen S_i desselben Fahrzeugs ist durch die Gleichung

$$P(S_t | S_{t-1}, A_{t-1}) \quad (3.4)$$

gegeben. Diese besagt, dass der Zustand S_t von seinem Vorgängerzustand S_{t-1} und der zum Zeitpunkt $t - 1$ durchgeführten Aktion A_{t-1} abhängt. Dem somit entstandenen dynamischen Bayes-Netz werden zusätzlich weitere Kanten hinzugefügt, die Auskunft über die Positionsrelationen der Fahrzeuge geben. Diese Kanten dürfen nur zwischen Zustandsknoten verschiedener Fahrzeuge zum selben Zeitpunkt existieren.

Ein weiterer sehr bedeutender Einfluss auf einen Zustand S_t ist die evt. zum gleichen Zeitpunkt akquirierte Beobachtung O_t . Die Abbildung 3.6 visualisiert diese geschilderten Zusammenhänge. Ein Vorteil dieses Netzes ist die Tatsache, dass trotz fehlender Beobachtungen O_i die Zustände anhand der Vorgängerzustände und der Aktionen geschätzt werden können. Ein wesentlicher Faktor liegt dabei in den Aktionen, die zu den jeweiligen Zeitpunkten den Fahrzeugen zugeordnet werden. Aktionen können dabei z. B. ein Spurwechsel, das Abbremsen, Beschleunigen, Abfahren von der Autobahn, etc. sein. Abhängig davon, welche Aktion man für ein Fahrzeug annimmt und in welcher topologischen Position es sich relativ zu den anderen Verkehrsteilnehmern befindet, kann der Folgezustand zum nächsten Zeitpunkt sehr genau ermittelt werden.

Ein Beispiel für eine solche topologische Karte ist in der Abbildung 3.7 gegeben. Dargestellt sind

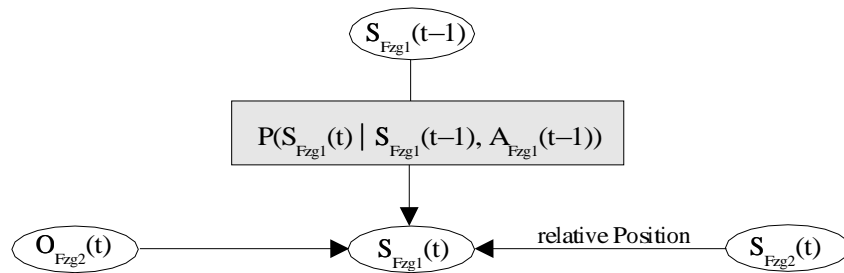


Abbildung 3.6: Einflüsse auf den Zustandsknoten S_t eines Fahrzeugs

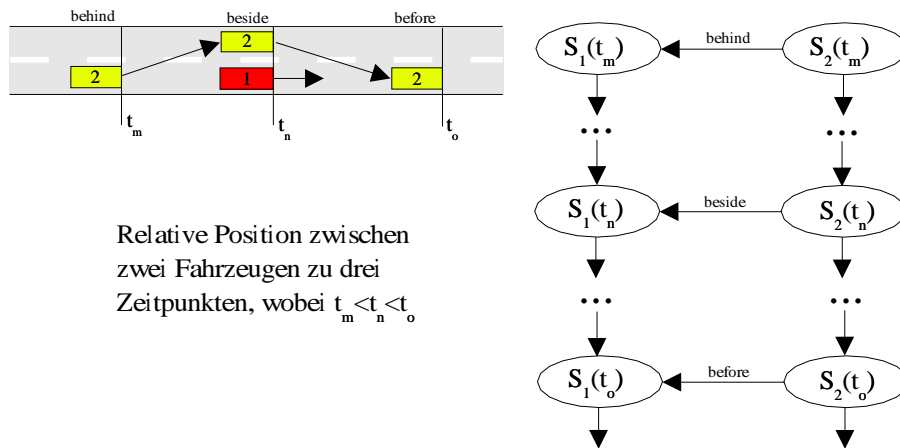


Abbildung 3.7: Beispiel einer topologischen Karte, die auf einem DBN basiert

drei Zeitpunkte aus der Sicht des Fahrzeugs 1. Zu jedem der Zeitpunkte befindet sich das beobachtete Fahrzeug 2 an einer anderen Position. Der daneben skizzierte Ausschnitt der zugehörigen topologischen Karte zeigt Spaltenweise die Zustände der 2 Fahrzeuge. Jede Zeile entspricht einem Zeitpunkt. Innerhalb jeder Zeile werden die Zustände der verschiedenen Fahrzeuge durch die relative Position verbunden. Damit sind in die Karte sowohl die zeitlichen, probabilistischen sowie topologischen Informationen integriert. Die Prädiktion, die Beobachtungen und die Zwischenzustände wurden in dem Beispiel aus Gründen der Übersichtlichkeit weggelassen.

Das in dieser Arbeit realisierte Weltmodell benutzt das hier vorgestellte probabilistische Netz mit topologischen Informationen für die Weiterführung, Prädiktion und Speicherung von Fahrzeugen. Mehr dazu ist im Kapitel 4.4.3 zu erfahren.

3.2 Anwendung

Zur prototypischen Implementierung der im vorherigen Kapitel betrachteten theoretischen Grundlagen wurden im Rahmen dieser Arbeit Untersuchungen vorgenommen, mit welcher Sensorik ein Fahrzeug ausgestattet werden könnte und welche Informationen über die Umwelt und das Fahrzeug selbst durch die Sensorik prinzipiell ermittelbar sind. Daraus lassen sich später das Sensornetzwerk, die möglichen Sensordatenfusionen, die im Weltmodell zu speichernden Merkmale und die Zuverlässigkeiten ableiten. Im Abschnitt 3.2.1 wird die Sensorik eines Fahrzeugs vorgestellt, wie sie in dieser Arbeit benutzt wurde. Danach wird im Abschnitt 3.2.2 eine Zusammenstellung gegeben, welche Merkmale durch die Sensorik prinzipiell geliefert werden könnten. Anschließend wird kurz auf die Möglichkeiten der Sensordatenfusionen eingegangen.

3.2.1 Mögliche Sensorausstattung des Versuchsfahrzeugs

Die Sensorausstattung war so zu konfigurieren, dass verschiedene Sensordatenfusionen möglich sind und der Sichtbereich nach vorn und hinten abgedeckt wird. Die Abbildung 3.8 stellt die externen Sensoren des Versuchsträgers dar. Anzumerken ist hierbei, dass der Versuchsträger während dieser Arbeit nur in der Simulation zur Verfügung stand. Hinter der Front- bzw.

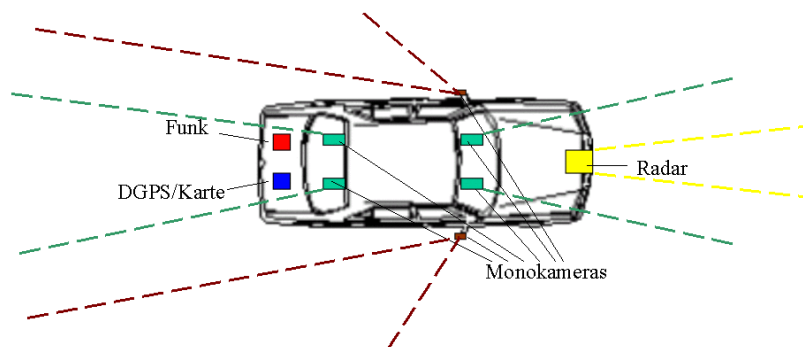


Abbildung 3.8: Versuchsträger mit 9 externen Sensoren

Heckscheibe sind jeweils zwei monokulare Kameras angebracht. Die Fusion der Bilder eines Kamerapaares ergibt ein Stereobild. Ein Kamerapaar wird deswegen auch als Stereokamera bezeichnet. Die vordere Stereokamera hat eine Sehweite von 60 m bei einem Öffnungswinkel von 36° . Die Stereokamera für den Heckbereich des Fahrzeugs hat eine größere Sehweite von 100 m, dafür beträgt der Öffnungswinkel nur 30° . Die Monokameras an den Spiegeln erweitern den Sehbereich des Fahrzeugs. Sie haben einen großen Öffnungswinkel von 40° . Die Sehbereiche aller Kameras sind verschieden, d. h. es gibt keine Überlappungen.

Am Kühlergrill befindet sich ein Radarsensor, der Objekte im Frontbereich des Fahrzeugs detektieren und deren Positionen und Geschwindigkeiten bestimmen kann. Der Radarwinkel beträgt 9° bei einer Reichweite von 150 m. Der Sehbereich des Radarsensors überlappt sich mit dem Sehbereich der vorderen Stereokamera. Dadurch werden einige Möglichkeiten für Fusionen der Radar- und Bildverarbeitungsdaten hergestellt. Der Radarsensor liefert im Gegensatz zu einer

Kamera bei schlechten Sichtverhältnissen (Dunkelheit, Regen, etc.) sehr gute Daten und wird daher als zusätzliche Sensordatenquelle gebraucht.

Über das DGPS-System und eine Karte soll die eigene Positionsbestimmung und der Erhalt von Straßenparametern erfolgen. DGPS steht für **Differential Global Positioning System**. Durch die Interpretation von empfangenen Signalen, die von Satelliten und von Stationen auf der Erde gesendet werden unter Berücksichtigung der Zeit kann die Position des Fahrzeugs metergenau bestimmt werden. Die Karte ist angedacht als Quelle für Informationen bzgl. der Straße (Verlauf, Anzahl der Spuren, Typ, etc.).

Zum Empfang von Funkinformationen anderer Fahrzeuge (CarTalk) wurde ein Funksensor vorgesehen. Die Kommunikation zwischen Fahrzeugen stellt eine zukünftige Herausforderung dar und wurde hier nur prinzipiell als Möglichkeit angedacht. Fahrzeuge könnten anderen ihre aktuelle Position, Geschwindigkeit, ihre momentanen Absichten oder ihr Fahrziel mitteilen.

3.2.2 Akquirierung von Informationen durch die Sensorik

Interessant ist nun die Frage, welche Informationen über die Umwelt und die Fahrzeugdaten des Versuchsträgers durch die beschriebene Sensorik und bereits existierende Algorithmen (z. B. Bildverarbeitung) geliefert werden können. Ein Überblick dazu wird in diesem Kapitel gegeben. Darüberhinaus wird aufgelistet, welche Merkmale in der Zukunft zusätzlich durch die Sensorik erkannt werden könnten, wobei die Zusammenstellung natürlich um weitere Sensoren und Merkmale erweiterbar ist. Die Gliederung erfolgt gemäß der Objektorientierung nach Klassen, wie sie auch in der Weltmodellierung verwendet werden:

- Straße mit ihren Spuren
- Eigene Fahrzeugdaten
- Objekte in der Umwelt
- Kontext
- StVO

Durch die zeilenweise Betrachtung der Tabellen erkennt man viele mögliche Fusionen von Sensordaten. Monokameras (M) und Stereokameras (St) sind zu einer Spalte zusammengefasst. “Sp l“ und “Sp r“ stehen für Spiegelkamera links bzw. rechts.

3.2.2.1 Straße

Die Straße setzt sich aus einzelnen Spuren zusammen. Diese werden separat im Abschnitt 3.2.2.2 behandelt. Zusätzlich zu den Attributen der einzelnen Spuren gibt es Merkmale, welche die Straße als Gesamtheit betreffen. Diese sind in der Tabelle 3.1 aufgeführt. Bei geteilten Autobahnen und Schnellstraßen (Grünstreifen in der Mitte) werden nur die Spuren einer Seite als Straße aufgefasst.

Merkmal	[]	M/St v	M/St h	Sp l	Sp r	GPS/Karte	Bemerkung
b	m	•	•			•	Gesamtbreite
n		•	•			•	Anzahl Spuren
Klasse		•				•	Autobahn, Stadtstraße, etc.
FahrbahnrandLi		○	○	○		○	Beschaffenheit (Breite, Belag, etc.)
FahrbahnrandRe		○	○		○	○	Beschaffenheit (Breite, Belag, etc.)
max. Höhe	m	○				•	z. B. für LKWs (Brücken, Tunnel)

M: Monokamera St: Stereokamera Sp: Spiegelkamera • vorhanden ○ prinzipiell möglich
 v: vorn h: hinten l: links r: rechts

Tabelle 3.1: Merkmale einer Straße

Die Breite der Straße und die Anzahl ihrer Spuren sind wichtige Informationen für die Planung von Überholvorgängen und Ausweichmanövern. Die Straßenklasse (Autobahn, Schnellstraße, Landstraße, Stadtstraße, etc.) setzt Bedingungen für das Verkehrsverhalten (Geschwindigkeit, Überholmanöver, etc.) und für die Suche nach Merkmalen in der Umwelt. So kann z. B. auf der Autobahn auf die Suche nach Fußgängerüberwegen und Kreuzungen in den Bildern der Stereokameras verzichtet werden.

Das Wissen über den Fahrbahnrand kann z. B. bei Ausweichmanövern lebensrettend sein. Die maximale Höhe, die Fahrzeuge auf einer Straße haben dürfen, ist insbesondere für Lastkraftwagen von Bedeutung.

Aus den Kamerabildern werden die Informationen über 2 Wege akquiriert:

- Erkennen der Merkmale direkt aus dem Bild (z. B. durch Markierungen)
- Erkennen von Verkehrszeichen, die Auskunft über die Straße geben (z. B. Autobahnschild, Wechsel von zwei- zu dreispuriger Autobahn, etc.)

Eine Karte kann prinzipiell alle Informationen über die Straße liefern. Problematisch ist hier jedoch die Aktualität, die Genauigkeit und die Menge der zu verwaltenden Daten.

3.2.2.2 Spur

Die Erkennung der Spurparameter ist eine der Grundvoraussetzungen für die automatische Quer- und Längsführung des Versuchsträgers. Auch für die räumliche Zuordnung von sich auf der Straße befindlichen Objekten ist das Wissen über die Spuren wesentlich. Um die Robustheit und Genauigkeit der Spurerkennung zu erhöhen, z. B. auch bei schlechten Sichtverhältnissen, ist es wichtig, alle verfügbaren Informationen über den Spurverlauf zu sammeln und in geeigneter Weise zu fusionieren (siehe [Zomotor97], [Gern2000]). Die Tabelle 3.2 gibt an, welche Sensoren bei der Spurerkennung mitwirken können und welche Merkmale von Interesse sind.

Merkm al	[]	M/St v	M/St h	Sp l	Sp r	Radar	Intern	GPS/Karte	Bemerkung
c0h	1/m	•	•			•		•	horiz. Krümmung
c1h	1/m	•	•			•		•	horiz. Krümmungsänderung
b	m	•	•					•	Breite
alpha	rad						○	○	Querneigung
c0v	1/m	○	○					○	vert. Krümmung
c1v	1/m	○	○					○	vert. Krümmungsänderung
GeoX	°						•	•	Geom. Breite
GeoY	°						•	•	Geom. Länge
h	m						•	•	Höhe über Meeresspiegel
X_{off}	m	•	•					•	Versatz von Straßenmitte
Oberfl.		○	○				○		Zustand (z. B. nass)
Schlagl.		○	○				○		Schlaglöcher
Mark. L		•	•	•				○	durchgezogen, gestrichelt
Mark. R		•	•		•			○	durchgezogen, gestrichelt

M: Monokamera St: Stereokamera Sp: Spiegelkamera • vorhanden ○ prinzipiell möglich
 v: vorn h: hinten l: links r: rechts

Tabelle 3.2: Merkmale einer Spur

Zu beachten ist bei dieser Auflistung, dass die Zuverlässigkeit der gelieferten Daten je nach Sensor und Sichtverhältnissen differiert. Die Radarinformation ist nur durch Fusionen mit anderen Sensorausgaben für die Spurerkennung von Nutzen (siehe [Gern2000]), d. h. allein durch das Radar kann keine Spur erkannt werden.

Die internen Sensoren können ebenso Informationen über die Spur liefern, z. B. die vertikale Krümmung (Steigung) oder die Querneigung.

Die aufgeführten Merkmale beziehen sich auf einen Punkt der Spur. Der Spurverlauf ergibt sich erst durch die Verkettung mehrerer solcher Punkte.

3.2.2.3 Eigene Fahrzeugdaten

Die Merkmale, die den Versuchsträger selbst betreffen, werden durch externe und interne Sensoren ermittelt. In der Tabelle 3.3 sind die Merkmale zur Beschreibung der kinematischen Werte (Geschwindigkeit, Position, etc.), der Ausmaße (Breite, Höhe, Länge), der Zustände der Teilsysteme (Blinker, Licht, etc.) und der weiteren Attribute (Gewicht, Stirnfläche, Fahrzeugtyp, etc.) des Versuchsträgers aufgelistet. Merkmale, deren Werte als unveränderlich angenommen werden können (z. B. Ausmaße, Typ, kinematische Maximalwerte), werden in einer separaten Spalte "Konstante" aufgeführt. Im Anhang B werden die Winkel des Fahrzeugs (Gierwinkel, Nickwinkel, Rollwinkel) anhand einer Skizze erklärt.

Merkmal	[]	M/St v/h	Sp r/l	Radar	GPS	Intern	Konstante	Bemerkung
V_{lat}	m/s	o	o		o	•		laterale Geschwindigkeit
V_{long}	m/s	o	o		•	•		longitudinale Geschw.
V_{max}	m/s						•	max. Geschwindigkeit
a_{lat}	m/s^2	o	o		o	•		laterale Beschleunigung
a_{long}	m/s^2	o	o		o	•		longitudinale Beschl.
a_{max}	m/s^2						•	max. Beschleunigung
$a_{maxVerz}$	m/s^2						•	max. Verzögerung
b	m						•	Breite
h	m						•	Höhe
l	m						•	Länge
Typ							•	Fahrzeugtyp (PKW, LKW)
GeoX	o				•	•		Geom. Breite
GeoY	o				•	•		Geom. Länge
Ψ_{NS}	rad					•		Winkel zur Nord-Süd-Achse
Spur		•	•		•			Fahrspurzuordnung
Xoff	m	•	•		o			Versatz von der Spurmitte
Ψ	rad	•	•	•				Gierwinkel zur Spurtangente
$\dot{\Psi}$	rad/s	•	•			•		Gierwinkeländerungsrate
α	rad	•	•			•		Nickwinkel
$\dot{\alpha}$	rad	o	o			•		Nickwinkeländerungsrate
β	rad	•	•			•		Rollwinkel
$\dot{\beta}$	rad	o	o			•		Rollwinkeländerungsrate
m_{Max}	kg						•	zulässiges Gesamtgewicht
m_{Ist}	kg					o		tatsächliches Gewicht
A_{Stirn}	m^2						•	Stirnfläche
Cw-Wert							•	Luftwiderstandsbeiwert
d_{Ro}							•	Rollwiderstandsbeiwert
Anhänger	1,0					•		ja/nein
Fahrtrichtung	v, r, 0					•		vorwärts, rückwärts, stehend
Lenkwinkel	rad					•		mom. Winkel des Lenkrades
Blinker						•		Status des Blinkers
Licht						•		Status des Fahrzeuglichts
Nebellicht						•		Status des Nebellichts
Bremslicht						•		Status des Bremslichts
Scheibenwischer						•		Status der Scheibenwischer
Gang						•		momentaner Gang
Motordrehzahl	1/min					•		momentane Drehzahl
Tempomat						•		mom. Wunschgeschw.

M: Monokamera St: Stereokamera Sp: Spiegelkamera • vorhanden o prinzipiell möglich
 v: vorn h: hinten l: links r: rechts

Tabelle 3.3: Eigene Fahrzeugdaten

Die Messung der Geschwindigkeit und Beschleunigung durch die interne Sensorik (z. B. Raddrehzahlmesser) erreicht bereits eine hohe Genauigkeit. Die Kameras könnten die Eigengeschwindigkeit/-beschleunigung z. B. durch Beobachtung fester Punkte der Straße (z. B. Markierungen oder Verkehrszeichen) über die Zeit ermitteln. Ebenso ist theoretisch die Schätzung der Geschwindigkeit und Beschleunigung durch den GPS-Sensor möglich, aber in der Praxis sind hier die Fehler noch zu groß.

Die Fahrzeugdaten haben für die Regelung des Fahrzeuges eine doppelte Bedeutung. Zum einen sind sie eine essentielle Information für den momentanen Zustand des Fahrzeugs, für die Situationsinterpretation und Prädiktion, ebenso wie die Umweltinformationen. Zum anderen sind die Werte der veränderlichen Fahrzeugmerkmale (z. B. V, a, Xoff) durch das Regelsystem direkt beeinflussbar, was einen Unterschied zu den Umweltmerkmalen darstellt.

3.2.2.4 Objekte in der Umwelt

Die Erkennung von Objekten in der Umwelt und deren Attributierung ist, wie bereits im Kapitel 2.1 erwähnt, ein Hauptbestandteil zukünftiger Fahrerassistenzsysteme. Dabei bekommt die Zuverlässigkeit, Vollständigkeit und Genauigkeit der Informationen einen sehr hohen Stellenwert, um das reale Verkehrsgeschehen durch das Weltmodell realitätsnah abbilden zu können.

Die Tabelle 3.4 enthält eine Sammlung von Merkmalen, die ein Objekt in der Umwelt beschreiben.

Merkmal	[]	M/St v/h	Sp r/l	Radar	Funk	Bemerkung
d_{lat}	m	•	◦	•		lateraler Abstand
d_{long}	m	•	•	•		longitudinaler Abstand
V_{lat}	m/s	•		•	•	laterale Geschwindigkeit
V_{long}	m/s	•	•	•	•	longitudinale Geschwindigkeit
a_{lat}	m/s^2	•		•	•	laterale Beschleunigung
a_{long}	m/s^2	•	•	•	•	longitudinale Beschleunigung
b	m	•			◦	Breite
h	m	•			◦	Höhe
l	m	•			◦	Länge
Typ		•			◦	Objektidentität (PKW, LKW, Fußgänger, etc.)
GeoX	◦	◦			•	Geom. Breite
GeoY	◦	◦			•	Geom. Länge
Spur		•	•		•	Fahrspurzuordnung
Xoff	m	•			◦	Versatz von der Spurmitte
Bewegungsrichtung	v, r, 0	•	•	•	•	vorwärts, rückwärts, stehend
Farbe		•	•		•	Farbe des Objekts
Blinker		•			•	Status des Blinkers
Licht		•			◦	Status des Fahrzeuglichts
Bremslicht		•			◦	Status des Bremslichts

M: Monokamera St: Stereokamera Sp: Spiegelkamera • vorhanden ◦ prinzipiell möglich
 v: vorn h: hinten l: links r: rechts

Tabelle 3.4: Merkmale eines Objekts in der Umwelt

Die Abstandsinformationen (d_{lat} und d_{long}) und die kinematischen Werte (V und a) sind hier relativ zum Versuchsträger zu interpretieren. Der Typ eines Objekts in der Umwelt ist jeder mögliche Gegenstand oder ein Lebewesen:

- Fahrzeug (PKW, LKW, Motorrad, Fahrrad, etc.)
- Verkehrszeichen, Ampel, Straßenmarkierung
- Fußgänger
- Tier (Reh, Wildschwein, Katze, etc.)
- Sonstiges

Aus Komplexitätsgründen wird in dieser Arbeit grundsätzlich davon ausgegangen, dass nur Fahrzeuge als Objekte in der Umwelt des Versuchsträgers vorkommen.

Durch den Funk werden nützliche Informationen von anderen Objekten selbst geliefert. Dabei ist zu beachten, dass die über den Funk übertragene Objektgeschwindigkeit und -beschleunigung absolut, dagegen die von den restlichen Sensoren gemessenen Werte relativ zum Versuchsträger zu interpretieren sind.

3.2.2.5 Kontext

Unter dem Kontext werden einige Merkmale der Umwelt des Versuchsträgers aufgelistet, die durch die Straße und die Objekte noch nicht abgedeckt wurden. Die folgende Tabelle enthält einige solcher weiteren Merkmale der Umwelt.

Merkm ^{al}	[]	M/St v	M/St h	Sp l/r	Intern	Karte	Bemerkung
Wetter		•	•	•	•		Regen, Schnee, Sonne, etc.
Licht		•	•	•	•		Tag, Nacht, Dämmerung, etc.
SichtVo	m	•					Sichtweite nach vorne
SichtHi	m		•				Sichtweite nach hinten
Umgebung		o	o	o		o	Bäume, Schluchten, Berge, etc.

M: Monokamera St: Stereokamera Sp: Spiegelkamera • vorhanden o prinzipiell möglich
v: vorn h: hinten l: links r: rechts

Tabelle 3.5: Merkmale des Kontextes

Das Wetter kann für ein Fahrerassistenzsystem eine wichtige Information z. B. zur Konfiguration des Sensornetzwerkes oder für Extraktionsmethoden, z. B. zur Spurerkennung, sein. Ebenso ist die Kenntnis über die Lichtverhältnisse z. B. zur Steuerung der Lichtanlage oder auch zur Auswertung von Kamerabildern nützlich. Das Wissen über die Sichtweite nach vorn und nach hinten (z. B. bei Nebel) gibt Aufschlüsse über die anzuratende Längsgeschwindigkeit, um im Notfall noch rechtzeitig bremsend oder ausweichend reagieren zu können. Informationen über die Beschaffenheit der Umgebung können ebenso brauchbar sein, indem z. B. durch das Wissen, dass sich Bäume am Straßenrand befinden, vom Radar erkannte Objekte besser zu klassifizieren sind.

Die Liste der Merkmale kann sicherlich noch durch weitere Merkmale ergänzt werden. An dieser Stelle soll dazu motiviert werden, auch solche Informationen über die Umwelt zukünftig in die Weltmodellierung zu integrieren und in den Methoden zu berücksichtigen.

3.2.2.6 StVO

Auch die Kenntnis über die aktuell gültigen Straßenverkehrsregeln ist bei zukünftigen Fahrerassistenzsystemen von Bedeutung. Dazu gehören generelle Vorschriften, die in einem Land gelten, aber auch die aktuellen durch die Verkehrsschilder, Markierungen und Ampeln angezeigten Ordnungen. Folgende Tabelle gibt an, welche Informationen durch welchen Sensor in diesem Zusammenhang ermittelt werden können.

Merkm ^{al}	M/St v	Karte	Konstante	Bemerkung
VZ	•	•		Verkehrszeichen (Typ, dLat, dLong)
Ampel	•	•		Zustand, dLat, dLong
Markierungen	•	o		Straßenpfeile, Fußgängerüberweg
generelle Geschw.-Beschr.		o	•	je nach Land verschieden
Vorfahrtsregeln			•	je nach Land verschieden

M: Monokamera St: Stereokamera v: vorn • vorhanden o prinzipiell möglich

Tabelle 3.6: Merkmale der Straßenverkehrsordnung

Die Verkehrszeichenerkennung auf Kamerabildern wurde bereits durch mehrere Arbeiten reali-

siert, ebenso wie die Erkennung von Ampeln und Markierungen (siehe [Janssen93], [Gavrila99]). Informationen über Markierungen oder Verkehrszeichen könnten prinzipiell auch in einer Karte gespeichert werden. Dabei stellen jedoch die Aktualität und die Datenmengen ein Problem dar.

3.2.3 Möglichkeiten der Sensordatenfusion

Durch die zeilenweise Betrachtung der im vorherigen Abschnitt abgebildeten Tabellen erkennt man, dass viele der Merkmale durch mehrere Sensoren akquiriert werden können. Dadurch sind viele Fusionen der Sensordaten möglich. Ein Beispiel dafür ist die Geschwindigkeit eines detektierten Objekts, die sowohl durch die Kameras, als auch durch das Radar ermittelt werden kann (siehe Tabelle 3.4). Befindet sich das Objekt gleichzeitig im Sehbereich einer Kamera und des Radars, könnte eine konkurrierende Fusion über zwei Geschwindigkeitsangaben stattfinden. Käme durch den Funk die von dem Objekt selbst gemessene Geschwindigkeit dazu, könnten sogar drei Geschwindigkeitsinformationen fusioniert werden. Neben der konkurrierenden Fusion ist die komplementäre Fusion von Bedeutung. Durch sie kann die Merkmalsliste eines Objekts vervollständigt werden. Wenn z. B. das Radar nur die kinematischen Merkmale eines Objekts liefert, eine Stereokamera dagegen nur die Ausmaße, können diese verschiedenen Merkmale für das Objekt zusammengefasst werden. Weiterhin ist die komplementäre Fusion durch die Verschmelzung der Sehbereiche zu einem großen Sehbereich gegeben. Z. B. durch das Hinzufügen von Spiegelkameras können zusätzliche Informationen über die Bereiche neben dem Fahrzeug ermittelt werden. Die Methoden für die verschiedenen Fusionen müssen nach [Levi2001] in einer Methodenbibliothek abgelegt werden. Je nach Sensornetzwerk und zu akquirierenden Merkmalen wird durch die Kontrollobjekte die benötigte Methodenauswahl getroffen.

3.3 Generische Klassenbibliothek für Fusionsvorgänge

Die in der Studie [Levi2001] vorgeschlagene Architektur zur dynamischen Konfiguration von Fusionsvorgängen (siehe Abschnitt 3.1.2) wurde vom **Forschungszentrum Informatik (FZI in Karlsruhe)** realisiert und stand für diese Arbeit als Klassenbibliothek zur Verfügung. Es handelt sich dabei um eine vom Anwendungsbereich unabhängige Bibliothek, die es ermöglicht, während der Laufzeit beliebige Funktionen und deren Ein- und Ausgabedaten in dynamisch veränderbarer Aufrufsreihenfolge zu verwalten. Dabei können die Ausgabedaten einer Funktion gleich den Eingabedaten einer weiteren Funktion sein. Trotz des generischen Charakters ist die Klassenbibliothek in erster Linie für die Konfiguration und Durchführung von Fusionsvorgängen vorgesehen. Zu Grunde liegt die Unterteilung eines Fusionsvorgangs in die verschiedenen Fusionsebenen, wie es in der Abbildung 3.1 im Abschnitt 3.1.1.3 gezeigt wird.

Dieses Kapitel stellt die Klassenbibliothek vor, indem zuerst die Klassen für die Steuerung der Fusionsvorgänge im Abschnitt 3.3.1 und die Klassen für die Ein- und Ausgabedaten der Fusionsmethoden im Abschnitt 3.3.2 erläutert werden. Anschliessend wird im Abschnitt 3.3.3 auf die Verwendung der Klassenbibliothek anhand eines Beispiels eingegangen. Die Verwendung der Klassenbibliothek in dieser Arbeit wird im Kapitel 4.3 im Rahmen der Realisierung des Fusionsagenten beschrieben.

3.3.1 Klassenhierarchie der Kontrollobjekte

Für die Steuerung der Fusionsvorgänge stehen die drei Klassen **Prozesskontrollobjekt**, **Fusionskontrollobjekt** und **Kontrollobjekt** zur Verfügung. Das Klassendiagramm in der Abbildung 3.9 zeigt deren wichtigsten Attribute, Methoden und Beziehungen. Die Klasse **Kontrollobjekt** bietet die notwendige Struktur für die Konfiguration einer einzelnen Fusions- oder Extraktionsmethode, kurz Methode genannt. Sie enthält eine Liste *input* für die Verweise auf die zu fusionierenden Eingabedaten. Auf die Ergebnisse der Methode wird durch die Einträge der Liste *output* verwiesen. Die Methode selbst wird durch den Funktionszeiger in *fusionsmethode()* referenziert. Zur Konfiguration und Ausführung der Methode stehen diverse Funktionen zur Verfügung, die nachfolgend beschrieben werden:

setFusion(*fusion)	Festlegen einer Methode, welche die Fusion durchführt
setExtraktion(*extraktion)	Festlegen einer Methode, durch die eine Extraktion vorgenommen wird
setParameter()	Deklaration der Slots, in denen die Objekte, die als Eingabedaten für die Methode dienen, stehen
setOutput()	Deklaration der Slots für die Ausgaben der Methode
starten()	Aufruf der Funktion <i>fusionsmethode()</i> , wobei die Objekte, die durch die Liste <i>input</i> referenziert sind, als Parameter übergeben werden

Die Anzahl der Ein- und Ausgabeobjekte ist variabel, jedoch kann pro Instanz der Klasse **Kontrollobjekt** höchstens **eine** Methode definiert werden.

Des Weiteren sind im Klassendiagramm 3.9 fünf von der Basisklasse **Kontrollobjekt** abgeleitete

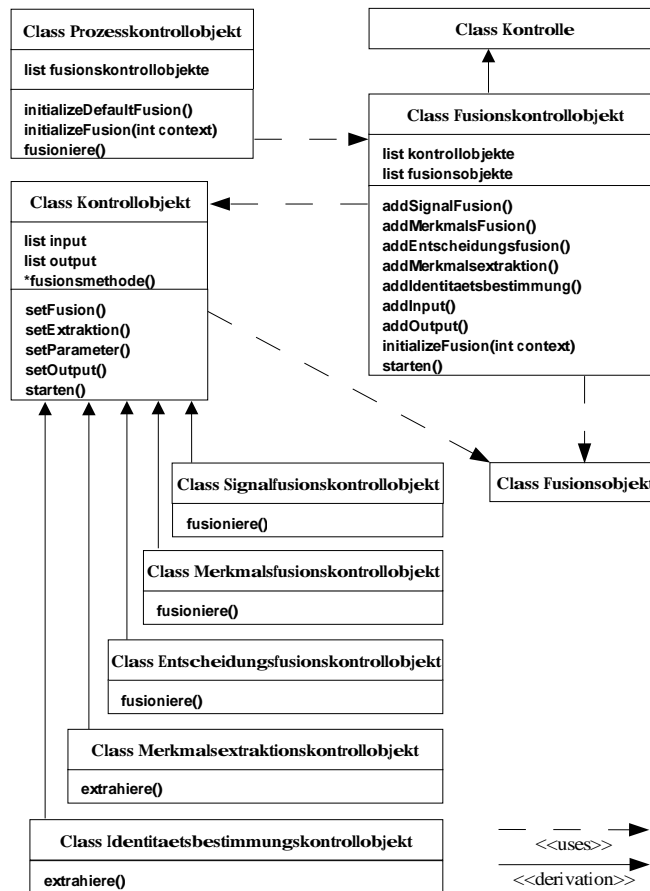


Abbildung 3.9: Klassendiagramm der Kontrollobjekte

Klassen zu sehen. Diese sind Spezialisierungen zur Trennung von Fusion, Extraktion und Identitätsbestimmung und innerhalb des Fusionsbereiches zur Unterscheidung der Fusionsebenen:

- Signalfusionskontrollobjekt** Klasse für die Konfiguration einer Fusion in der Signalfusionsebene
- Merkmalsfusionskontrollobjekt** Klasse für die Konfiguration einer Fusion in der Merkmalsebene
- Entscheidungsfusionskontrollobjekt** Klasse für die Konfiguration einer Fusion in der Entscheidungsebene
- Merkmalsextraktionskontrollobjekt** Klasse für die Konfiguration einer Methode, die Merkmale extrahiert; dieser Vorgang wird hauptsächlich auf der Signalebene durchgeführt
- Identitaetsbestimmungskontrollobjekt** Klasse für die Konfiguration einer Methode, die anhand von Merkmalen eine Identitaetsbestimmung vornimmt; dieser Vorgang ist der Merkmalsebene zuzuordnen

Durch die Funktionen *fusioniere()* und *extrahiere()* kann die jeweilige Methode alternativ zu *starten()* aufgerufen werden.

Die Konfiguration eines kompletten Fusionsvorganges unter Berücksichtigung **mehrerer** Methoden und Fusionsdaten wird mit Instanzen der Klasse **Fusionskontrollobjekt** möglich. Dabei kann ein Fusionsvorgang gemäß 3.1.1.3 strukturiert werden, indem alle Fusionsebenen durchlaufen werden und die Ausgabe einer Ebene gleich der Eingabe der darüberliegenden Ebene ist. Die Klasse **Fusionskontrollobjekt** hält die Methoden, aus denen sich der Fusionsvorgang aufbaut, in der Liste *kontrollobjekte*, wobei eine Methode durch ein Objekt der zuvor vorgestellten Klasse **Kontrollobjekt** beschrieben wird. Neben dem geordneten Aufruf der einzelnen Methoden ist auch die Konfiguration der Methoden, d. h. die Zuordnung von Ein- und Ausgabedaten durch die Klasse **Fusionskontrollobjekt** realisierbar, wodurch von der Klasse **Kontrollobjekt** abstrahiert werden kann. Die Ein- und Ausgabedaten der Methoden werden in der Liste *fusionsobjekte* gehalten. Folgende Zusammenstellung beschreibt Funktionen, die zur Konfiguration und zum Starten eines Fusionsvorganges verwendet werden können:

addSignalfusion()	Eingabeparameter: <ul style="list-style-type: none">• Funktionszeiger auf die auszuführende Signalfusionsmethode• Referenzen auf die Eingabedatenlots der Signalfusionsmethode• Anzahl der Ausgabedaten• Referenz auf den Aufrufer der Fusionsmethode
addMerkmalsfusion()	Als Resultat wird eine Liste der Referenzen auf die Ausgabedatenlots zurückgegeben.
addEntscheidungsfusion()	analog zu <i>addSignalfusion()</i> unter Angabe einer Merkmalsfusionsmethode
addMerkmalsextraktion()	analog zu <i>addSignalfusion()</i> unter Angabe einer Merkmalsextraktionsmethode
addIdentitaetsbestimmung()	analog zu <i>addSignalfusion()</i> unter Angabe einer Identitaetsbestimmungsmethode
declareInput()	Anfordern eines Slots, in dem ein Eingabedatum hinterlegt werden kann
declareOutput()	Anfordern eines Slots, in dem ein Ausgabedatum hinterlegt werden kann
initializeFusion(int context)	konfigurieren eines Standardfusionsvorganges unter Angabe eines Kontextes
starten()	sequentieller Aufruf aller definierten Kontrollobjekte nach der vorgegebenen Reihenfolge

Ein Beispiel für die Verwendung der Klasse **Fusionskontrollobjekt** ist im Abschnitt 3.3.3 gegeben. Die Klasse **Prozesskontrollobjekt** schließlich ist für die Verwaltung der Objekte der Klasse

Fusionskontrollobjekt zuständig. Sie hält die Referenzen auf alle ihr zugehörigen Fusionskontrollobjekte und bietet folgende Funktionen an:

- initializeDefaultFusion()** Erzeugung eines Standardfusionskontrollobjekts
- initializeFusion(int context)** Erzeugung eines neuen Fusionskontrollobjekts unter Angabe eines evt. vordefinierten Fusionsvorgangs
- fusioniere()** Der Fusionsprozess wird durchgeführt, indem alle Fusionskontrollobjekte ihren Fusionsvorgang starten

3.3.2 Klassenhierarchie der Fusionsobjekte

Die Klassen aller Ein- und Ausgabedaten der auszuführenden Fusionsmethoden **müssen** von der Klasse **Fusionsobjekt** **abgeleitet** sein. Dadurch wird eine allgemeine Handhabung der Daten möglich. Die Abbildung 3.10 zeigt die zu Grunde liegende Klassenstruktur.

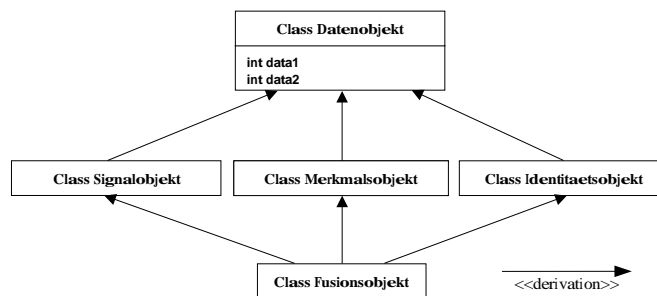


Abbildung 3.10: Klassendiagramm der Fusionsobjekte

In der Klasse **Datenobjekt** können beliebige Attribute und Methoden definiert werden, die für alle Fusionsobjekte zur Verfügung stehen sollen. Die Klassen **Signalobjekt**, **Merkmalsobjekt** und **Identitaetsobjekt** sind Spezialisierungen, die je nach Anwendungsbereich der Klassenbibliothek für die jeweilige Fusionsebene angepasst werden können (siehe Abschnitt 3.1.3). Die bereits erwähnte Klasse **Fusionsobjekt** ist durch die Mehrfachvererbung in der Lage, Objekte zu generieren, die jeder der Fusionsebenen angehören können.

3.3.3 Verwendung der Klassenbibliothek

Wie Fusionsvorgänge mit der Klassenbibliothek konfiguriert und durchgeführt werden können, wird in diesem Abschnitt mit Hilfe eines ausführlichen Beispiels demonstriert, das alle Fusionsebenen umfasst. Zuerst folgt eine sprachliche und bildliche Beschreibung des Fusionsvorganges, danach wird anhand von Codefragmenten in der Syntax von C++ die Verwendung der Klassenbibliothek erläutert.

Der als Beispiel dienende Fusionsvorgang ist im Kontext eines Fahrzeuges zu sehen, das mit einem Radarsensor und zwei Monokameras an der Frontseite bestückt ist, mit dem Ziel, eine Hinderniserkennung durchzuführen. Die Bilder der Monokameras werden auf der *Signalebene* miteinander zu einem Stereobild fusioniert. Auf diesem Stereobild wird eine *Merkmalsextraktion* vorgenommen, um Merkmale von potenziellen Hindernissen zu akquirieren. Z. B. wird dabei ein Bildbereich

detektiert, der auf ein Objekt schließen lässt. Dessen Breite, Höhe und Abstand von der Kamera werden berechnet. Parallel dazu wird für die vom Radarsensor erkannten Objekte eine Liste von Merkmalen erzeugt. Diese Liste enthält mit großer Wahrscheinlichkeit auch die Merkmale des mit dem Stereobild detektierten Objekts (z. B. Abstand, Winkel und Geschwindigkeit). Auf der *Merkmalsebene* werden die Merkmale des Stereobildes und des Radarsensors fusioniert, wobei diese gemeinsamen Objekten zugeordnet werden. Eine solche Fusion kann in diesem Beispiel das Zuordnen der Breite, der Höhe, des Abstandes und der Geschwindigkeit zu einem neu erzeugten Fusionsobjekt sein, wobei der Abstand z. B. Ergebnis einer Mittelung von Stereo- und Radarinformation ist. Über die in der Merkmalsebene entstandenen gemeinsamen Objekte wird eine *Identitätsbestimmung* vorgenommen, für die zwei Verfahren zur Verfügung stehen. Das erste Verfahren könnte z. B. aus den genannten Merkmalen auf einen PKW schließen, das zweite dagegen auf einen LKW. Die Ergebnisse der Verfahren werden auf der *Entscheidungsebene* wiederum fusioniert, wobei eine Entscheidung für die Identität eines jeden Objekts gefällt wird, z. B. die Entscheidung für PKW. Das Resultat des gesamten Vorgangs sind einzelne *Fusionsobjekte*, die an die Weltmodellierung weitergegeben werden können. Die Abbildung 3.11 stellt den beschriebenen Fusionsvorgang nochmals dar.

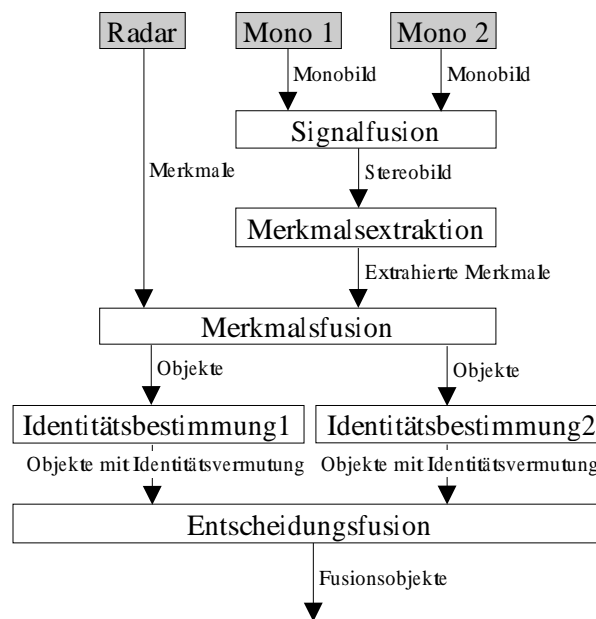


Abbildung 3.11: Beispielhafter Fusionsvorgang

Für die Implementierung des Fusionsvorganges muss ein Prozesskontrollobjekt existieren, das die Kontrolle über alle Fusionsvorgänge übernimmt. Dieses kann durch

```
FUSION.Prozesskontrollobjekt *pko = new FUSION.Prozesskontrollobjekt();
```

vorgenommen werden. Über das Prozesskontrollobjekt *pko* ist es nun möglich, ein Fusionskontrollobjekt zu erzeugen, das den oben beschriebenen Fusionsvorgang steuert:

```
FUSION_Fusionskontrollobjekt *fko = pko->FUSION_initializeFusion(EMPTY_FUSION);
```

Dieses neue, leere Fusionskontrollobjekt *fko* bekommt als nächstes mitgeteilt, in welchen Slots die Eingabedaten für die zuerst stattfindende Signalfusion zu finden sind. Dafür werden zuerst zwei neue Slots deklariert und deren Nummern in zwei Variablen zwischengespeichert:

```
int inputMono1 = fko->FUSION_declareInput();  
int inputMono2 = fko->FUSION_declareInput();
```

Dies bedeutet, dass im Slot, der durch den Wert in “inputMono1“ beschrieben wird, das Bild der einen Monokamera stehen wird. Analog dazu ist in “inputMono2“ das Bild der zweiten Monokamera zu finden.

Die Konfiguration einer Methode wird mit der Angabe der Fusionsebene, dem Funktionszeiger auf die zu verwendende Methode und einer Liste mit den Eingabedatenslots vollzogen. Die Liste der Eingabedatenslots kann einfach erzeugt werden durch:

```
vector <int> parameter;  
parameter.push_back(inputMono1);  
parameter.push_back(inputMono2);
```

Nun kann die Signalfusionsmethode in den Fusionsvorgang eingefügt werden, wobei die Fusionsebene, der Zeiger auf die auszuführende Methode und die Liste der Slots, in denen später bei der Ausführung der Methode die Eingabedaten zu finden sind, angegeben werden müssen:

```
int outputSignalfusion = fko->FUSION_addFusion(SIGNALFUSION, &fuseMonoImages, parameter);
```

Dabei wird der Slot zurückgegeben, in dem das Resultat der Fusion, das Stereobild, zu finden sein wird. Die Methode “fuseMonoImages“ steht hier für eine Fusionsmethode, die Stereobilder aus zwei Monobildern erzeugen kann.

Als nächstes findet die Merkmalsextraktion auf dem fusionierten Stereobild, das aus dem Slot *outputSignalfusion* ausgelesen werden kann, statt. Für diese Aufgabe wird eine weitere Methode nach demselben Muster deklariert:

```
vector <int> parameter;  
parameter.push_back(outputSignalfusion);  
int outputMerkmalsextraktion = fko->FUSION_addFusion(MERKMALSEXTRAKTION, &extractMerkmale,  
parameter);
```

Das Ergebnis der Methode “extractMerkmale“ wären in unserem Beispiel während der Ausführung des Fusionsvorganges die Merkmale Breite, Höhe und Abstand des erkannten Objekts.

Die nächste Ebene ist die *Merkmaleebene*. Auf dieser werden die Radar- und Stereodaten fusioniert. Der Slot für die Radardaten muss neu deklariert werden, ebenso die Fusionsmethode:

```
vector <int> parameter;  
int inputRadar = fko->FUSION_declareInput();
```

```
parameter.push_back(outputMerkmalsextraktion);  
parameter.push_back(inputRadar);  
int outputMerkmalfusion = fko->FUSION_addFusion(MERKMALFUSION, &fuseStereoAndRadar, parameter);
```

Im Slot *inputRadar* sind in unserem Beispiel die Merkmale Abstand, Geschwindigkeit und Winkel des erkannten Objekts zu finden. Diese werden mit den Merkmalen Breite, Höhe und Abstand, die von der oben beschriebenen Merkmalsextraktion in den Slot *outputMerkmalsextraktion* geschrieben werden, durch die Methode *fuseStereoAndRadar* fusioniert, wobei ein neues Fusionsobjekt angelegt wird. Dieses wird in den Slot *outputMerkmalfusion* geschrieben.

Nun folgt eine Algorithmenfusion, da auf dem gleichen Ausgabedatum der *Merkmalsfusionsebene* verschiedene Methoden zur Identitätsbestimmung der Objekte angesetzt und deren Resultate fusioniert werden. Die Identitätsbestimmungsmethoden lassen sich folgendermaßen konfigurieren:

```
vector<int> parameter;  
parameter.push_back(outputMerkmalfusion);  
int outputIdentBest1 = fko->FUSION_addFusion(IDENTITAETSBESTIMMUNG, &IB1, parameter);  
int outputIdentBest2 = fko->FUSION_addFusion(IDENTITAETSBESTIMMUNG, &IB2, parameter);
```

Beide Methoden bekommen die Ausgabe der Merkmalsfusion, in unserem Beispiel ein Objekt mit den Merkmalen Breite, Höhe, Abstand und Geschwindigkeit, als Eingabe. Durch die Verschiedenheit der Algorithmen liefert *IB1* z. B. die Hypothese “PKW“ und *IB2* die Hypothese “LKW“. Diese Hypothesen werden wiederum in Slots geschrieben.

Zuletzt werden die Ergebnisse der Identitätsbestimmungsalgorithmen in der *Entscheidungsebene* fusioniert:

```
vector<int> parameter;  
parameter.push_back(outputIdentBest1);  
parameter.push_back(outputIdentBest2);  
int outputEntscheidung = fko->FUSION_addFusion(ENTSCHEIDUNGSFUSION, &decide, parameter);
```

Die Methode “decide“ entscheidet sich für eine der Hypothesen. Im Slot *outputEntscheidung* sind schließlich die Fusionsobjekte als Endergebnisse des Fusionsvorganges zu finden, in unserem Beispiel ein Objekt mit den Merkmalen Breite, Höhe, Abstand, Geschwindigkeit und Identität PKW. Nachdem in der beschriebenen Weise der Fusionsvorgang konfiguriert wurde, können die Eingabedatenslots *inputRadar*, *inputMono1* und *inputMono2* mit Werten gefüllt werden:

```
fko->FUSION_fillInput(inputRadar, (FUSION_Fusionsobjekt*) radarListe);  
fko->FUSION_fillInput(inputMono1, (FUSION_Fusionsobjekt*) monoBild1);  
fko->FUSION_fillInput(inputMono2, (FUSION_Fusionsobjekt*) monoBild2);
```

Gestartet wird der Fusionsvorgang durch das Prozesskontrollobjekt, welches wiederum durch

```
pko->FUSION_fusioniere();
```

in Gang gebracht wird. Das Endergebnis des gesamten Fusionsvorgangs kann mit

```
foListe = fko->FUSION_getOutput(outputEntscheidung);
```

gelesen werden.

Nach einmaliger Konfiguration des Fusionsvorgangs müssen lediglich bei jedem neuen Zyklus die Eingabedaten neu gefüllt und der Fusionsprozess gestartet werden. Als Eingabedaten können dabei ebenso Ausgabedaten eines vorherigen Zyklus dienen. Bei Bedarf ist es auch möglich, die Konfiguration des Fusionsvorgangs dynamisch zu ändern.

Kapitel 4

Entwurf und Implementierung

Zur praktischen Erprobung der in [Levi2001] gemachten Vorschläge zur Strukturierung der Fusionsvorgänge und zum Weltmodell (siehe Kapitel 3.1) wurde im Rahmen dieser Arbeit die Implementierung eines Beispielsystems durchgeführt. Trotz des prototypischen Charakters wurde auf eine erweiterbare, über das Beispielsystem hinaus verwendbare Struktur Wert gelegt.

Das Beispielsystem stellt eine Teilmenge eines Fahrerassistenzsystems dar (siehe Abschnitt 2.1), wie im Bild 4.1 zu sehen ist. Es werden Informationen aus der Umwelt eines Fahrzeugs durch dessen Sensorik aufgenommen, was durch die Box "Sensoren" ausgedrückt wird. Diese Informationen werden in der Box "Fusionen" in geeigneter Weise fusioniert. Die Resultate der Fusionen werden in der Box "Weltmodell" in eine konsistente Repräsentationsstruktur gebracht, auf der die Situationsinterpretation aufsetzen kann.

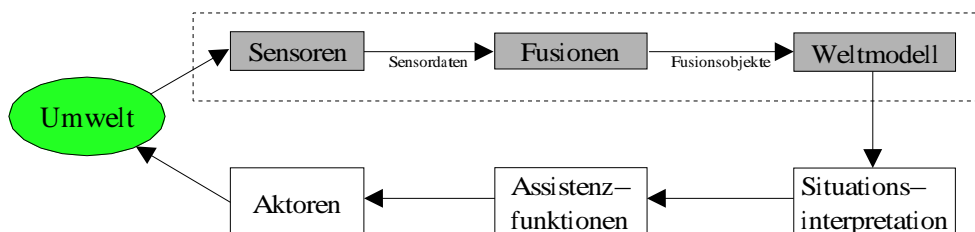


Abbildung 4.1: Einbettung des zu entwickelnden Systems

Das Beispielsystem wurde unter Verwendung von ANTS (siehe Abschnitt 2.3) realisiert. Die genaue Beschreibung der agentenbasierten Softwarearchitektur des Beispielsystems ist im Abschnitt 4.1 gegeben.

Zur Akquirierung der Sensordaten standen mehrere Möglichkeiten zur Auswahl. Eine Möglichkeit wäre ein realer Versuchsträger gewesen. Aufgrund der Ungewissheit, ob während dieser Arbeit ein solcher zur Verfügung stehen wird und weil die Herstellung der Szenarien, die für den Test notwendig sind, auf öffentlichen Straßen aufwendig ist, wurde nach Alternativen gesucht. Eine der Alternativen war das Lesen aus manuell konstruierten oder von Versuchsträgern aufgezeichneten Dateien. Die dritte Möglichkeit war die Integration einer vorhandenen Simulation. Die Entscheidung fiel für die Simulation, da man mit ihr flexibel Verkehrsszenen herstellen kann,

die Generierung von Sensordaten möglich ist und die Erweiterungen zur Verwendbarkeit im hier vorgestellten System mit geringem Aufwand machbar zu sein schienen. Die Simulation wird im Abschnitt 4.2 vorgestellt.

Die Fusion der Sensordaten wurde unter Verwendung einer vorhandenen Klassenbibliothek (siehe Abschnitt 3.3) in einem von den Sensoren und dem Weltmodell separierten Fusionsagenten realisiert. Dessen Struktur und Logik wird im Abschnitt 4.3 geschildert. Der Weltmodellagent, der auf den Fusionsresultaten aufsetzt, ist ebenso ein separates Modul. Dessen Realisierung ist im Abschnitt 4.4 nachzulesen.

4.1 Softwarearchitektur

Die Struktur des Systems ist durch die bereits oben erwähnte Unterteilung in die Bereiche

- Akquirierung von Umweltinformationen,
- Fusion der Informationen und
- konsistente Repräsentation der erkannten Umwelt

grob vorgegeben. Die Abbildung 4.2 zeigt eine Verfeinerung dieser Struktur unter Angabe des Datenflusses und der Hauptaufgaben der Fusions- und Weltmodellagenten. Der Datenfluss zwi-

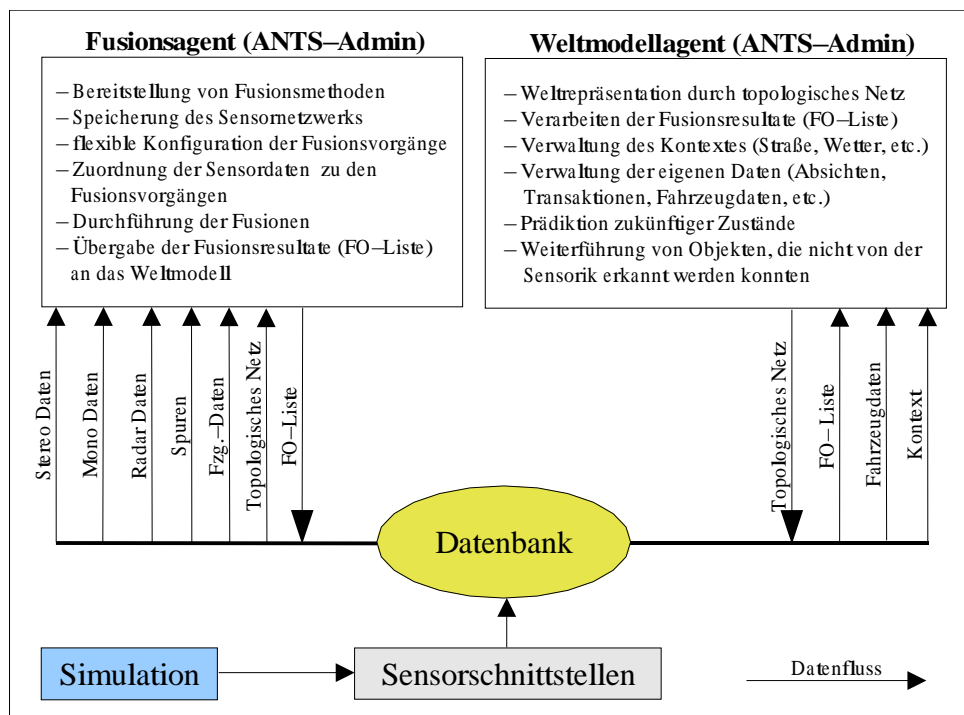


Abbildung 4.2: Architektur des Systems

schen den Einheiten des Systems erfolgt über eine globale ANTS-Datenbank. Die Schnittstellen zu den Sensoren holen die Daten von den Sensoren und schreiben diese in die Datenbank. Der Fusionsagent liest zu Beginn seines Zyklus die aktuellen Sensordaten und, falls vorhanden, die jüngste Version der Weltrepräsentation in Form eines topologischen Netzes aus der Datenbank ein. Nach der Durchführung der Fusionsvorgänge legt der Fusionsagent die Fusionsresultate in der Datenbank ab. Diese werden vom Weltmodellagenten gelesen und verarbeitet. Am Ende des Zyklus des Weltmodellagenten wird das nun gültige konsistente topologische Netz in die Datenbank geschrieben, damit es anderen Agenten, die es brauchen, zugänglich ist.

Gründe für das Separieren der Fusions- und Weltmodellagenten sind vor allem die damit gegebenen Möglichkeiten der Parallelisierung sowie Austauschbarkeit der Module. Ein Nachteil ist der nötige Datentransfer zwischen den Agenten über die Datenbank.

Die Integration der Sensorschnittstellen und der Simulation in das ANTS-System wurde so entworfen, dass die Simulation mit akzeptablem Aufwand durch einen realen Versuchsträger oder eine andere Sensordatenquelle ersetzt werden kann. Die Sensorschnittstellen werden durch Funktionseinheiten realisiert, die von den Sensoren die Sensordaten holen und diese an die globale ANTS-Datenbank weitergeben. Durch eine Parametrisierung wird den Funktionseinheiten mitgeteilt, wie sie die Sensordaten beschaffen können. In dieser Arbeit wurden die Kommunikationsschnittstellen zu den Sensoren ausschließlich mit der Simulation parametrisiert.

Somit ist es in Zukunft möglich, das gleiche System im Labor mit einer Simulation zu testen, das auch in einem Versuchsträger lauffähig ist, wobei lediglich die Parametrisierung der Sensorschnittstellen angepasst werden muss, um die Sensordatenquelle anzugeben. Die Abbildung 4.3 zeigt diese Struktur.

Die Architektur des Gesamtsystems ist beliebig erweiterbar. Es können sowohl neue Agenten, als

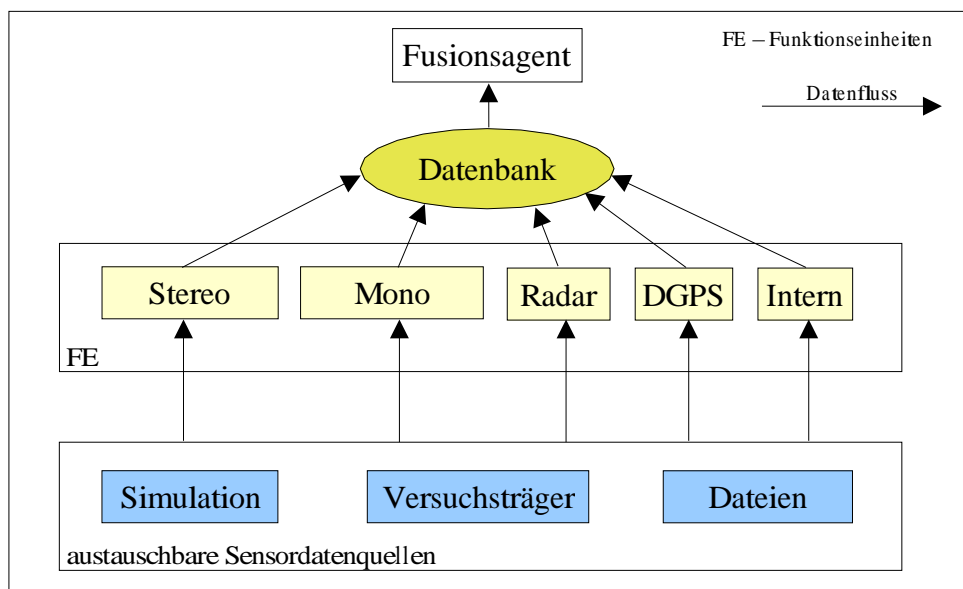


Abbildung 4.3: Funktionseinheiten zur Beschaffung der Sensordaten

auch zusätzliche Kommunikationswege hinzugefügt werden.

Die Zykluszeiten der einzelnen Sensoren und Agenten sind voneinander unabhängig. Weiterhin ist

es durch ANTS mit sehr geringem Aufwand möglich, das System verteilt über mehrere Rechner zu konfigurieren.

Der nächste Abschnitt geht auf die unterste Ebene des Systems, die Simulation als Sensordatenquelle, sowie auf die Schnittstelle zu den Sensordaten ein.

4.2 Simulationsumgebung

Dieses Kapitel beschäftigt sich mit der Simulation, die für diese Arbeit zur Verfügung stand und zur Verwendbarkeit angepasst werden musste. Im Abschnitt 4.2.1 wird die Simulation kurz vorgestellt. Im nachfolgenden Abschnitt 4.2.2 sind die in dieser Arbeit durchgeführten Erweiterungen nachzulesen. Der Abschnitt 4.2.3 beschreibt die Funktionseinheiten, die für den Transfer der Sensordaten in die globale Datenbank zuständig sind.

4.2.1 Simulation

Die Simulation wurde ursprünglich konzipiert, um Kamerasequenzen einer virtuellen Autofahrt zu generieren. Die Autofahrt findet auf einer vom Benutzer definierten Straße statt, auf der beliebig viele weitere Fahrzeuge positioniert werden können. Zusätzlich zu den Kamerabildern stehen GPS- und Radardaten zur Verfügung.

Die Sensordaten werden schrittweise erzeugt. Die Schrittweite lässt sich vor dem Simulationslauf einstellen, ebenso die Anzahl der Simulationsschritte. Der Straßenverlauf wird aus einer Datei ausgelesen. Die Anzahl, Breite und Helligkeit der Fahrspuren, Randstreifen und Markierungen ist entlang der Straße dynamisch änderbar und es können Kurven und Steigungen definiert werden. Der Initialzustand (Position, Geschwindigkeit, Verhalten) und die Ausmaße der Fahrzeuge auf der Straße (einbezogen das eigene) werden ebenfalls in der Parameterdatei hinterlegt und beim Start der Simulation eingelesen. Für das Verhalten der Fahrzeuge während des Simulationslaufs stehen 4 verschiedene Fahrprofile zur Verfügung:

- Geradeausfahrt:** Fahrzeug fährt mit konstantem Offset zur Spurmitte
- Schlingern:** Fahrzeug fährt eine "Schlangenlinie"
- Nicken:** Fahrzeug nickt (z. B. wie nach Fahrt über eine Bodenwelle)
- Spurwechsel:** Fahrzeug fährt benutzerdefinierten Offset nach links bzw. rechts

Zur Generierung der Bilder enthält die Simulation eine virtuelle Stereokamera, deren Blickrichtung nach vorn ist. Die Parameter einer solchen virtuellen Kamera (z. B. Brennweite, Einbauwinkel, Höhe über der Straße, etc.) entsprechen den Parametern einer realen Kamera. Die Stereokamera besteht aus zwei nebeneinander positionierten Monokameras, deren Blickfeld sich überlappt. Das Bild jeder Monokamera enthält den sichtbaren Straßenverlauf und die Rückseiten der vorausfahrenden Fahrzeuge.

Die Radardaten werden aus den Positionen der vorausfahrenden Fahrzeuge ermittelt. Die GPS-Datensätze werden aus dem Straßenverlauf erzeugt, wobei unter Angabe eines Fehlerquotienten eine Verrauschung der Daten bewirkt werden kann.

Zum Betrachten der generierten Bilder und zur Ausgabe von Werten (Krümmung der Spur, Geschwindigkeiten, etc.) steht ein Fenster zur Verfügung, in dem die Daten des vorherigen Simulationsschritts angezeigt werden. In diesem wird auch die aktuelle Verkehrsszene aus der Vogelperspektive

spektive dargestellt, um dem Benutzer einen Überblick über die Situation zu geben. Im nächsten Kapitel wird erläutert, welche Erweiterungen im Rahmen dieser Arbeit an der Simulation durchgeführt wurden.

4.2.2 Erweiterungen der Simulation

Die im Abschnitt 4.2.1 vorgestellte Simulation wurde in dieser Arbeit angepasst und erweitert, um als Sensordatenquelle verwendet werden zu können. Das Verhalten der Fahrzeuge musste vordefiniert sein, um das Erzeugen von Verkehrsszenen zu ermöglichen, wie sie im Abschnitt 5.1 gezeigt werden. Es wurden eine Stereokamera mit der Blickrichtung nach hinten und zwei Rückspiegelkameras als weitere Sensoren benötigt, um die geplante Sensorausstattung, wie sie in 3.2.1 nachzulesen ist, zur Verfügung zu haben. Diese Erweiterung wird im Abschnitt 4.2.2.2 näher beleuchtet. Weiterhin war es neben der Erzeugung der Kamerabilder der neuen Sensoren notwendig, die "beobachtbare Leistung" der Sensoren und Bildverarbeitungsalgorithmen nachzubilden. Mehr dazu ist im Abschnitt 4.2.2.3 zu finden.

Die Simulation wurde von dem oben erwähnten Fenster zur Anzeige der Bilder und Interaktion mit dem Benutzer für diese Arbeit entkoppelt und in einen ANTS-Server (siehe Kapitel 2.3) transformiert. Bei jedem Simulationsschritt werden die aktuellen Sensordaten generiert und den Sensorschnittstellen zum Abholen bereitgestellt.

4.2.2.1 Verhalten der Fahrzeuge

Bisher konnten die Fahrprofile der Fahrzeuge um den Versuchsträger herum nur einmalig durch die Parameterdatei zugeordnet werden. Die hinterlegten Profile wurden ab dem Start endlos abgefahren, es sei denn, es handelte sich um die Aktion eines Spurwechsels, die nach dem Vollzug des Spurwechsels beendet ist. Für das eigene Fahrzeug war es allerdings durch Interaktion mit dem Benutzer während des Simulationslaufs bereits möglich, das Verhalten dynamisch zu ändern.

Zur Beschreibung des Verhaltens eines Fahrzeugs wurde eine **Aktionsliste** eingeführt. Jede der Aktionen beschreibt den Start oder das Ende eines Fahrprofils. Der Ausführungszeitpunkt der Aktion wird durch die Angabe der Länge der Strecke, die bis zur Ausführung der Aktion abgefahren werden muss, festgelegt. Bei jedem Simulationsschritt wird überprüft, ob eine Aktion der Aktionsliste ausgeführt werden kann.

Der Benutzer hat die Möglichkeit, die Aktionen in einer Datei zu hinterlegen. Dadurch können beliebig viele verschiedene und komplexe Verkehrsszenen angelegt werden, auf die jederzeit Zugriff besteht. Die Abbildung 4.4 gibt ein Beispiel für die Aktionsliste, die das Verhalten eines überholenden Fahrzeugs beschreibt.

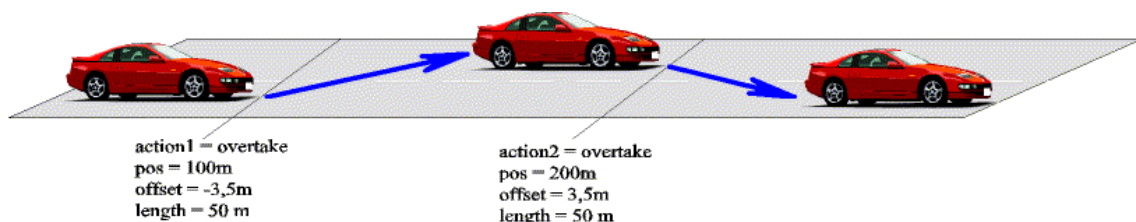


Abbildung 4.4: Aktionsliste für einen Überholvorgang

4.2.2.2 Zusätzliche Sensoren

Für einen erweiterten Sichtbereich wurden vier weitere Monokameras in den Simulator eingebaut. Zwei Monokameras decken den Sichtbereich nach hinten ab und ergeben zusammen eine Stereokamera. Die zwei restlichen Monokameras wurden an den Rückspiegeln angebracht, um die Sicht in den "toten Winkel" zu ermöglichen. Die Abbildung 4.5 zeigt die neuen Kameras und deren Sichtbereich. Durch das Hinzufügen der beschriebenen Kameras entstand das Problem des Re-

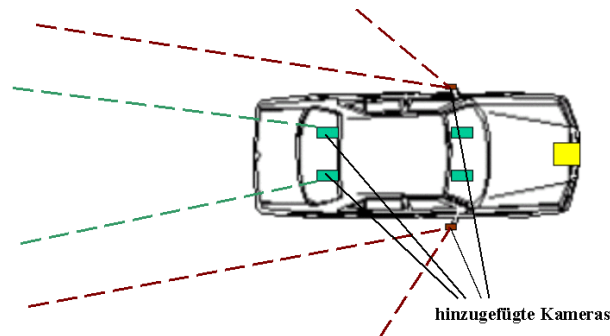


Abbildung 4.5: Hinzugefügte Sensoren

chenaufwandes für das Generieren der Bilder zu jedem Simulationsschritt. Damit wäre die Simulation als Sensordatenquelle für das in dieser Arbeit zu entwickelnde System nur bedingt brauchbar gewesen. Desweiteren war es zu aufwendig, die bestehenden Bildverarbeitungsalgorithmen auf die erzeugten Bilder anzusetzen. Aus diesen Gründen fiel die Entscheidung, die Sensorausgaben in Form der "beobachteten Leistung" bereitzustellen, da dies Rechen- und Entwicklungszeit spart. Das Generieren der Bilder kann dabei weiterhin, über Parametrisierung einstellbar, erfolgen. Auf die beobachtbare Leistung geht das nächste Kapitel ein.

4.2.2.3 Beobachtbare Leistung

Als "beobachtbare Leistung" wird die Ausgabe von realen Sensoren oder von Algorithmen bezeichnet, die durch künstlich erzeugte Daten nachgebildet wird. Diese erzeugten Daten sind so realitätsnah wie möglich, damit die Weiterverarbeitung der Daten unabhängig von deren Erzeugung erfolgen kann und durch deren nichtidealisierten Werte die wirkliche Leistung von Sensoren bzw. Algorithmen nicht verloren geht.

Im Rahmen dieser Arbeit wurden alle am Versuchsfahrzeug installierten Kameras durch die "beobachtbare Leistung" ersetzt. Somit ist die Ausgabe einer Kamera bereits die Liste aller im aktuellen Bild zu findenden Merkmale. Dadurch besteht keine Abhängigkeit von bestehenden Merkmalsextraktionsalgorithmen, die in das zu entwickelnde System hätten integriert werden müssen. Nachfolgend wird angegeben, welche Merkmale die Simulation als Sensorausgabe zur Verfügung stellt. Die Stereokameras vorn und hinten geben jeweils folgende Merkmale der erkannten Objekte aus:

- VLong:** relative Längsgeschwindigkeit
- VLat:** relative Quergeschwindigkeit
- DistLong:** Längsabstand (zwischen den Fahrzeugnullpunkten)
- DistLat:** Seitlicher Versatz
- Length:** Länge des erkannten Objekts
- Width:** Breite des erkannten Objekts
- TimeStamp:** Zeitpunkt, zu dem die Merkmalswerte gelten

Die Monokameras an den Spiegeln liefern analog zu den Stereokameras die relative Längsgeschwindigkeit, den Längs- und Querabstand der erkannten Objekte und den Zeitpunkt, für den die Werte gültig sind.

Zur Realisierung der beobachtbaren Leistung werden alle im Simulationslauf existierenden Objekte bezüglich ihres Erscheinens im Sichtbereich der Sensoren geprüft. Sind sie im Sichtbereich eines Sensors, werden die oben aufgeführten Merkmale ermittelt und in die Ausgabeliste des Sensors eingefügt.

Der Straßenverlauf, die Radardaten und die eigenen Fahrzeugdaten werden ebenfalls direkt aus der Simulation gelesen und als Sensorausgabe zur Verfügung gestellt.

Im Folgenden werden die Schnittstellen zu den Sensordatenquellen beschrieben.

4.2.3 Schnittstellen zur Simulation realer Sensoren

Wie bereits im Abschnitt 4.1 beschrieben, sind die Schnittstellen zu den Sensoren so konzipiert, dass die Sensordatenquellen leicht ausgetauscht werden können und für die Klienten, welche die Schnittstellen nutzen, ein einheitlicher Zugriff auf die Sensordaten ohne das Wissen über deren ursprüngliche Herkunft möglich ist. Die Grobstruktur einer solchen Schnittstelle wird in der Abbildung 4.6 gezeigt. Der Teil der Kommunikation zu den realen Sensoren bzw. zur Simulation

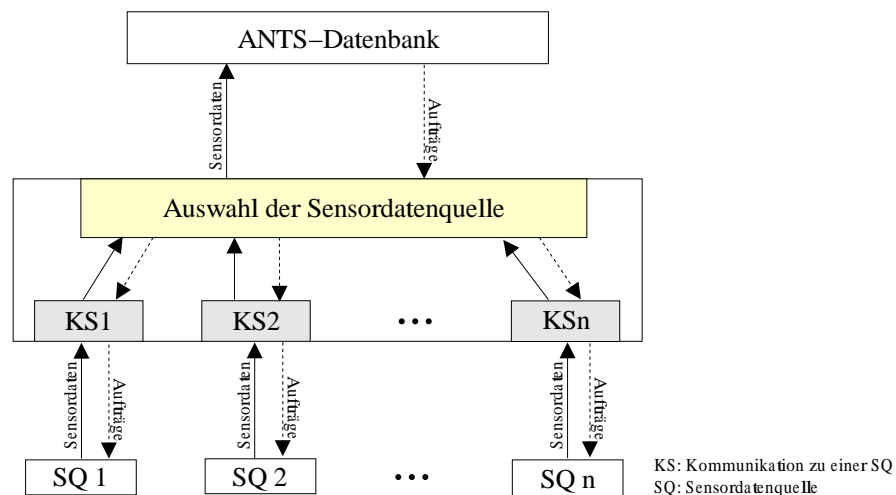


Abbildung 4.6: Grobe Struktur einer Sensorschnittstelle

muss bei jeder Schnittstelle individuell programmiert werden. Wichtig ist das einheitliche Datenformat der Sensordaten, in das die Sensorausgabe der Sensordatenquelle konvertiert werden muss. Der Teil, der die Sensordaten an die globale ANTS-Datenbank verschickt, ist wiederum für alle Schnittstellen einheitlich.

Die Struktur lässt es auch zu, den umgekehrten Weg zu gehen. Sollen Informationen an die Sensordatenquelle gesandt werden (z. B. Änderung der Konfiguration), können diese zuerst an die Sensorschnittstelle in einheitlichem Format weitergegeben werden. Diese ist dafür verantwortlich, in der "korrekten Sprache" die Information an die jeweilige Sensordatenquelle weiterzuleiten.

In der ANTS-Architektur ist jede Schnittstelle zu den Sensordaten eine Funktionseinheit. Somit haben die Agenten Kontrolle darüber, welche Funktionseinheit zu welcher Zeit aktiv ist.

In dieser Arbeit wurden Funktionseinheiten, d. h. Sensorschnittstellen für die Stereo- und Monokameras, für die Radardaten, für den Straßenverlauf und für die eigenen Fahrzeugdaten realisiert. Für jede Art von Sensordaten wurde ein eigenes ANTS-Datenbankobjekt entworfen. Die Datenbankobjekte werden in jedem Zyklus der Funktionseinheiten mit Sensordaten gefüllt und an die globale ANTS-Datenbank gesandt. An dieser Stelle kommt nun der im nächsten Kapitel beschriebene Fusionsagent in Spiel, der seine Arbeit auf diesen Sensordaten aufbaut.

4.3 Fusionsagent

Der in dieser Arbeit entwickelte Fusionsagent hat zur Aufgabe, die von den verschiedenen Sensoren gelieferten Objektbeschreibungen in einheitliche Fusionsobjekte zu transformieren und diese dem Weltmodell zur Verfügung zu stellen. Dabei wird das aktuelle Weltmodell zur Zuordnung der Daten zu bereits bekannten Objekten in die Fusionsvorgänge einbezogen. Das Bild 4.7 verdeutlicht diesen Vorgang.

Die realisierten Fusionsvorgänge beinhalten nur die Merkmals- und Entscheidungsebene, da die

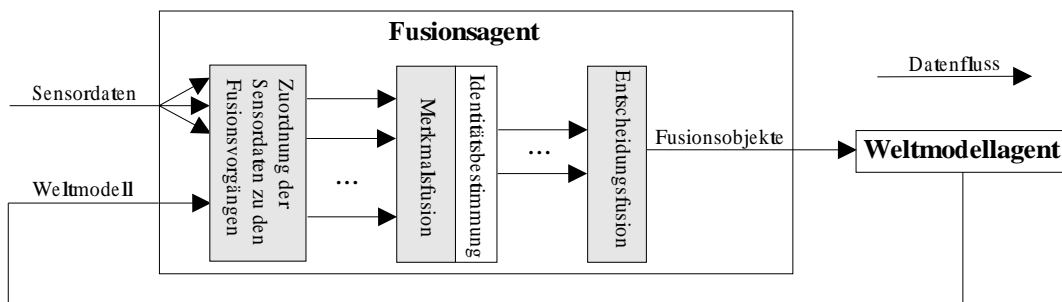


Abbildung 4.7: Grobe Struktur des Fusionsagenten

Sensorausgaben bereits extrahierte Merkmale sind und damit die Signalebene übersprungen wird. Des Weiteren beschränken sich die Fusionsvorgänge auf die Generierung von Fusionsobjekten für die erkannten Objekte und das eigene Fahrzeug.

Die Arbeitsweise des Fusionsagenten wird im Abschnitt 4.3.1 beschrieben. Die Verwendung der Kontrollobjekte zur Beschreibung von Fusionsvorgängen ist im Abschnitt 4.3.2 nachzulesen. Danach werden einige der in dieser Arbeit implementierten Fusionsmethoden im Abschnitt 4.3.3 vorgestellt.

4.3.1 Zyklus des Fusionsagenten

Der Fusionsagent ist eine autonome Einheit. In ANTS bedeutet dies, dass der Zyklus, in dem die Berechnungen des Agenten erfolgen, endlos wiederholt aufgerufen wird. Innerhalb eines solchen Zyklus arbeitet der Agent die ihm aufgetragenen Aufgaben ab.

Zu Beginn des Zyklus des Fusionsagenten liest dieser die aktuellen Sensordaten und das Weltmodell aus der globalen ANTS-Datenbank ein. Die Sensordaten sind in diesem System extrahierte Merkmale, die von jedem Sensor bereits zu Objekten zugeordnet wurden. Für die Erkennung von Fahrzeugen in der Umwelt stehen dem Fusionsagenten solche Objektlisten vom Radarsensor, von zwei Spiegelkameras (links und rechts) und zwei Stereokameras (vorn und hinten) zur Verfügung. Weiterhin werden vom GPS-Sensor der aktuelle Straßenverlauf in Form eines Objekts, das vorverarbeitete Daten enthält, und von der internen Sensorik die eigenen Fahrzeugdaten gelesen. Das aktuelle Weltmodell ist in der Datenbank in Form eines topologischen Netzes zu finden.

Die nächste Aufgabe des Fusionsagenten besteht darin, herauszufinden, welche Objekte aus den Sensordaten die Beschreibungen desselben realen Objekts in der Umwelt sind, damit deren Merkmale fusioniert werden können. Des Weiteren muss mit Hilfe des aktuellen Weltmodells überprüft

werden, ob ein Objekt neu erkannt wurde oder es bereits im Weltmodell gehalten wird. Die bereits bekannten Objekte werden ebenso zur Fusion der Sensordaten herangezogen, um Informationen aus der Vergangenheit nutzen zu können. Für jedes in der Umwelt erkannte Objekt wird in diesem Zusammenhang ein separater Fusionsvorgang konfiguriert. Wird ein reales Objekt im aktuellen Zyklus neu erkannt, wird dafür ein neuer Fusionsvorgang angelegt. Dagegen wird bei einem bereits bekannten Objekt der schon konfigurierte Fusionsvorgang wiederverwendet und gegebenenfalls neuen Gegebenheiten angepasst. Der Algorithmus zur Realisierung der Zuordnung der Sensordaten zu ihren Fusionsvorgängen ist im Teilabschnitt 4.3.3.1 nachzulesen.

Nachdem alle Sensordaten den Fusionsvorgängen als Eingabe zugewiesen sind, wird der Fusionsprozess gestartet. Dabei sind die Abarbeitungen der einzelnen, für die Objekte in der Umwelt angelegten und vom Fusionsprozess aufgerufenen Fusionsvorgänge voneinander unabhängig und lassen eine parallele Ausführung zu. In dieser Implementierung wurden die Fusionsvorgänge jedoch sequentiell abgearbeitet, da die Möglichkeit der Parallelisierung in der benutzten Klassenbibliothek (Abschnitt 3.3) nicht gegeben ist. Wie die Fusionsvorgänge konfiguriert wurden, ist im Abschnitt 4.3.2 beschrieben.

Nachdem die Fusionsvorgänge durchlaufen wurden, gibt der Fusionsagent die Resultate in Form von Fusionsobjekten in einer Liste dem Weltmodell weiter, indem er sie in die globale ANTS-Datenbank schreibt. Zusammenfassend wird nochmal der Zyklus des Fusionsagenten dargestellt:

1. Lesen der aktuellen Sensordaten aus der globalen ANTS-Datenbank
2. Zuordnung der Sensordaten zu den Fusionsvorgängen
3. Konfiguration der Fusionsvorgänge
4. Ausführung der Fusionsvorgänge
5. Schreiben der Fusionsobjekte als Ergebnis der Fusionsvorgänge in die globale ANTS-Datenbank

Detaillierte Informationen über die implementierten Fusionsvorgänge sind in den folgenden Abschnitten zu finden.

4.3.2 Verwendung der Kontrollobjekte

Das in der Studie [Levi2001] vorgeschlagene Konzept, Fusionsvorgänge zu strukturieren, wurde in dem in dieser Arbeit entwickelten Fusionsagenten mit Hilfe der im Abschnitt 3.3 vorgestellten Klassenbibliothek umgesetzt und getestet. Dem Fusionsagenten steht ein Prozesskontrollobjekt zur Verfügung, mit dem er alle Fusionsvorgänge verwalten kann. Die Fusionsvorgänge werden durch Fusionskontrollobjekte beschrieben. Bei der Initialisierung des Fusionsagenten wird das Prozesskontrollobjekt angelegt und dabei ein Standardfusionsvorgang erzeugt. Dieser Standardfusionsvorgang ist u. a. dafür verantwortlich, die eingehenden Sensordaten den Fusionsvorgängen zuzuordnen und für neu erkannte Objekte in der Umwelt neue Fusionsvorgänge zu erzeugen. Im Standardfusionsvorgang finden folgende Fusionen statt:

- Spurzuordnung des eigenen Fahrzeugs durch Fusion von Fahrzeug- und Straßendaten (siehe 4.3.3.3)

- Fusion des aktuellen Weltmodells mit den neuen Sensordaten (siehe 4.3.3.1) um Herauszu- finden, welche Objekte dem System bereits bekannt sind

Somit wird die Logik des Prozesskontrollobjekts durch ein Fusionskontrollobjekt ausgedrückt. Die Konfiguration dieses Standardfusionskontrollobjektes lässt sich während der Laufzeit ändern und ist abhängig vom aktuellen Kontext (Garage, Stadtstraße, Autobahn, etc.) und der Verfügbarkeit der Sensordaten. Die Initialkonfiguration kann beim Erzeugen des Prozesskontrollobjektes durch die Angabe des Kontextes vorgegeben werden. In dieser Arbeit wurde die Konfiguration für den Kontext *Autobahn* implementiert.

Alle weiteren Fusionskontrollobjekte sind jeweils genau einem Objekt in der Umwelt zugeordnet. Wenn ein beispielsweise ein Fahrzeug durch das Radar neu erkannt wird, ist es die Aufgabe des Standardfusionskontrollobjektes, dafür einen neuen Fusionsvorgang zu konfigurieren. Dieser neue Fusionsvorgang bekommt eine Identifikationsnummer zugeteilt, unter der das erkannte Fahrzeug in Zukunft innerhalb des Fahrerassistenzsystems verwaltet wird. Die Sensordaten, die das Objekt betreffen, werden in jedem Zyklus vom Standardfusionsvorgang dem zugehörigen Fusionskontrollobjekt durch das Schreiben in sogenannte Eingabedatenslots zugeteilt. Der Fusionsvorgang, der auf den Informationen, die in den Eingabedatenslots zu finden sind, aufsetzt und die Sensordaten zu einem Fusionsobjekt verschmilzt, ist in der Abbildung 4.8 zu sehen. Die Eingabedatenslots

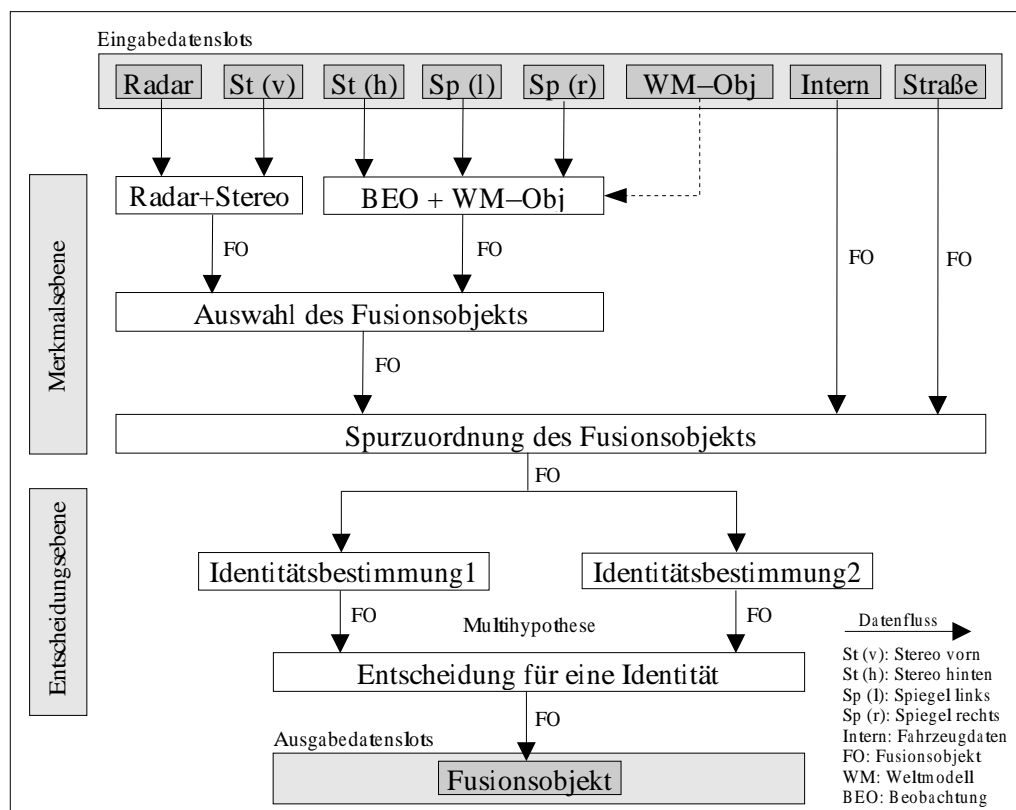


Abbildung 4.8: Fusionsvorgang für ein Objekt in der Umwelt

müssen in jedem Zyklus durch das Standardfusionskontrollobjekt gefüllt werden. Die Slots, für

die keine Daten zur Verfügung stehen, bleiben leer. Abhängig davon, welche Eingabeslots gefüllt sind, werden die Fusionsmethoden durchgeführt. Sind die Slots *Radar* und *St* (v) gefüllt, wird die Fusionsmethode zur Fusionierung der Radar- und Stereomerkmale ausgeführt. Ist nur die Radarinformation oder nur die Stereoinformation über das vor dem Fahrzeug befindliche Objekt vorhanden, wird anhand der jeweils gegebenen Daten trotzdem ein Fusionsobjekt erstellt. Sind weder Radar- noch Stereodaten der vorderen Kamera vorhanden, wird die Fusionsmethode nicht ausgeführt. Eine solche Fusionsmethode ist immer dann zu benutzen, wenn sich die Sehbereiche von Sensoren überlappen, um die besten Merkmale aus allen Informationen herausfiltern und dem Fusionsobjekt anheften zu können.

Die restlichen Sensoren (Stereokamera hinten und die Spiegelkameras) haben in diesem System keine überlappenden Sehbereiche und dienen nur der Vergrößerung des Gesamtsehbereiches des Fahrzeugs. Ein Datum, das nur in einem der Slots *St* (h), *Sp* (l) und *Sp* (r) steht, wird auch in ein Fusionsobjekt konvertiert, um ein einheitliches Datenformat für die Übergabe des Objektes in das Weltmodell zu schaffen. Diese Konvertierung erfolgt unter Zuhilfenahme der Informationen, die über das Objekt evt. schon im Weltmodell bekannt sind und ist somit auch eine Fusionsmethode. Ein Objekt, das dem Weltmodell bereits bekannt ist, wird in jedem Zyklus seinem korrespondierenden Fusionskontrollobjekt im Slot *WM-Obj* bereitgestellt. Beim Überholen eines langen LKW könnte es z. B. vorkommen, dass sowohl die rechte Spiegelkamera, als auch die hintere Stereokamera Teile des LKW erkennen. In diesem Fall würden in den Slots *Sp* (l) und *St* (h) Daten zu finden sein, die verschiedene Teile des LKW beschreiben. Die Methode *BEO + WM-Obj* ist dafür verantwortlich, daraus ein einheitliches Fusionsobjekt zu erzeugen.

Als nächstes wird in den Ausgabedatenslots der eben beschriebenen Methoden nach dem Fusionsobjekt gesucht, das weiterverarbeitet werden soll. Es wird davon ausgegangen, dass nur in einem der Slots ein Fusionsobjekt zu finden ist, da ausgeschlossen wird, dass ein einzelnes Objekt gleichzeitig durch die Sensorik für den Frontbereich und die Sensorik für den Bereich hinter dem Fahrzeug erkannt werden kann. Wurde in einem Zyklus das Objekt, für das der Fusionsvorgang durchgeführt wird, von keinem der Sensoren detektiert, endet der Fusionsvorgang an dieser Stelle. Wurde ein Fusionsobjekt zur Weiterverarbeitung gefunden, wird anhand der zusätzlichen Informationen über das eigene Fahrzeug (Slot *Intern*) und die Straße eine Spurzuordnung vorgenommen. Danach durchläuft das Fusionsobjekt zwei Identitätsbestimmungsmethoden, in denen die Identität (z. B. PKW, Motorrad, etc.) in Form einer Hypothese vorgeschlagen wird (Erzeugung einer Multihypothese). Für eine der Hypothesen entscheidet sich eine weitere Fusionsmethode, was der Algorithmenfusion entspricht, bei der auf einem Datensatz verschiedene Methoden angewandt werden. Das nun berechnete Fusionsobjekt als Endresultat des Fusionsvorgangs wird im Ausgabeslot *Fusionsobjekt* abgelegt.

4.3.3 Fusionsmethoden

In diesem Abschnitt werden einige der implementierten Fusionsmethoden vorgestellt. Zu bemerken ist hierbei, dass die Methoden prototypischen Charakter haben und in dieser Arbeit der Test der Fusionsarchitektur im Vordergrund steht. Die Methoden sind in einer Klasse *FusionFunctions* untergebracht. Diese Klasse übernimmt die Aufgabe der in der Studie [Levi2001] genannten Methodenbibliothek.

Eine für den Fusionsagenten sehr wichtige Methode wird im Abschnitt 4.3.3.1 vorgestellt. Sie übernimmt die Zuordnung der Sensordaten zu den Fusionskontrollobjekten und deren Verwaltung

unter Berücksichtigung des aktuellen Weltmodells. Im Abschnitt 4.3.3.2 wird kurz auf die Fusion der Merkmale von Radar- und Stereoobjekten eingegangen. Algorithmen für die Spurzuordnung der Fahrzeuge sind in den Abschnitten 4.3.3.3 und 4.3.3.4 zu finden.

4.3.3.1 Erkennen neuer und bekannter Objekte und deren Zuordnung zu den Fusionsvorgängen

Für jedes reale Objekt in der Umwelt existiert ein korrespondierendes Fusionskontrollobjekt im Fusionsagenten. Benötigt wird nun eine Logik, durch die alle ankommenden Sensordaten auf den richtigen Weiterverarbeitungsweg gelenkt werden. Es muss überprüft werden, ob das Objekt in der Umwelt, zu dem Merkmale akquiriert wurden, bereits dem System bekannt ist. Ist dies der Fall, existiert dafür ein Fusionskontrollobjekt. Diesem werden die Merkmale als Eingabe zugewiesen. Wenn das durch die Sensorik detektierte Objekt dem System noch nicht bekannt ist, muss dafür ein neues Fusionskontrollobjekt angelegt werden. Wie diese Logik im Rahmen dieser Arbeit implementiert wurde, wird nachfolgend beschrieben.

Zur Verfügung stehen die aktuellen Merkmalslisten aller Sensoren, in denen die Merkmale bereits Objekten in der Umwelt zugeordnet sind. Weiterhin ist die Referenz auf das Weltmodell in Form eines mit einem Zeitstempel versehenen topologischen Netzes gegeben. Der Algorithmus für die oben beschriebene Logik enthält folgende Aktionen:

1. Löschen der Eingabedaten des letzten Zyklus aller Fusionskontrollobjekte, die für ein Objekt in der Umwelt zuständig sind
2. Durchgehen aller Merkmalslisten, die von den Sensoren gesandt wurden; dabei wird für jedes Objekt Obj_{Sensor} angestoßen:
 - (a) Durchsuchen des topologischen Netzes nach einem Objekt Obj_{WM} , dessen Position unter Einbeziehung eines Toleranzbereiches der Position von Obj_{Sensor} entspricht
 - (b) wurde Obj_{WM} gefunden, wird über dessen Identifikationsnummer das zugehörige Fusionskontrollobjekt gesucht und diesem die Beobachtung Obj_{Sensor} in einen vorgesehenen Eingabeslot zugewiesen
 - (c) wurde Obj_{WM} nicht gefunden, wird Obj_{Sensor} als neu erkanntes Objekt in einer Liste vorgemerkt
3. alle Objekte Obj_{Sensor} der verschiedenen Sensoren, die als neue Objekte vorgemerkt wurden, werden anhand der metrischen Position in der Umwelt zu einer Einheit zusammengefasst und für jede Einheit wird ein neues Fusionskontrollobjekt angelegt
4. Konfiguration des im Abschnitt 4.3.2 vorgestellten Fusionsvorgangs für jedes neu erstellte Fusionskontrollobjekt

Nach der Durchführung der angegebenen Schritte sind im aktuellen Zyklus des Fusionsagenten alle Sensordaten den zugehörigen Fusionskontrollobjekten zugeordnet worden.

Informationen zur Erprobung und zu Problemen des Algorithmus und zukünftige Erweiterungsmöglichkeiten sind in den Kapiteln 5 und 6.2.1 zu lesen.

4.3.3.2 Stereo und Radar

Die Fusion der Sensordaten, die von der vorderen Stereokamera und vom Radarsensor stammen, ist für einige Merkmale konkurrierend. Grund dafür ist der gemeinsame Sehbereich der Sensoren und die gemeinsamen Merkmale, wie z. B. der *Abstand* und die *Längsgeschwindigkeit*. Andere Attribute, wie z. B. die Fahrzeugausmaße, können nur durch die Stereokamera bestimmt werden und bilden damit komplementäre Information, die im Fusionsobjekt hinterlegt wird.

In dieser Arbeit werden die Stereodaten zuverlässiger als die Radardaten eingestuft. Dementsprechend werden für die konkurrierenden Merkmale die Stereodaten den Radardaten vorgezogen. Eine weitere Möglichkeit wäre z. B. die Mittelung der Werte gewesen. Nähme man an, dass der durch den Radarsensor ermittelte longitudinale Abstand zuverlässiger als der gelieferte Abstandswert von der Stereokamera wäre, aber der laterale Abstand als Resultat einer Stereobildauswertung genauer als der vom Radar berechnete Detektionswinkel ist, könnte man die jeweils zuverlässigsten Merkmale den Sensordaten entnehmen und im Fusionsobjekt vereinigen.

Im Rahmen dieser Arbeit hatte die Methode vordergründlich die Aufgabe, zwei Sensordatensätze zu einem zu verschmelzen.

4.3.3.3 Spurzuordnung des eigenen Fahrzeugs

Das Zuordnen des eigenen Fahrzeugs zu einer Spur ist erst möglich, wenn sowohl die aktuellen Fahrzeug- als auch die Straßendaten zur Verfügung stehen. Dies ist in dem hier beschriebenen System erst im Fusionsagenten der Fall. Voraussetzung für den Algorithmus sind:

- die Kenntnis über die gesamte Straße (Anzahl der Spuren, Spurenbreiten und Positionen der Spuren)
- das Wissen, welchen lateralen Abstand das eigene Fahrzeug von der Mitte der gesamten Straße hat

Sind diese Daten bekannt, kann die Spur, auf der sich das Fahrzeug befindet, durch einfache Berechnung ermittelt werden. Als Fusion wird die Methode hier deshalb aufgeführt, da sie die Informationen zweier Sensordatensätze nutzt, um eine neue Information, in diesem Falle die Spurnummer des eigenen Fahrzeugs, zu erzeugen.

4.3.3.4 Spurzuordnung eines Objekts in der Umwelt

Die Eingaben dieser Methode sind ein Fusionsobjekt, dessen Spur berechnet werden soll, das eigene Fahrzeug, das bereits einer Spur zugeordnet wurde und die Straßendaten. Diese Informationen sind nur im Fusionsagenten und im Weltmodellagenten vorhanden. Durch das eigene Fahrzeug und dessen lateralen Abstand vom Objekt in der Umwelt wird der Offset von der Straßenmitte ermittelt. Anhand der Straßendaten und dem Offset von der Spurmitte kann dem Fusionsobjekt nun eine Spur zugeordnet werden.

4.4 Weltmodellagent

Der Weltmodellagent ist dafür zuständig, das aktuelle Wissen über die Umwelt- und Eigendaten des Versuchsträgers in einer konsistenten Repräsentation zur Verfügung zu stellen. Das Wissen wird aus den fusionierten Sensordaten, die der Fusionsagent liefert, abgeleitet. Zu den Umweltdaten gehören Informationen über die Objekte um das Fahrzeug herum, die Straßendaten, das Wetter, die Lichtverhältnisse, die Verkehrszeichen, etc., die das Verhalten des Fahrzeugs und die Konfiguration des Sensornetzwerks beeinflussen können. Die eigenen Daten beinhalten die Fahrzeugdaten, Absichten, Erkennungsaufgaben und Transaktionen des Versuchsträgers [Levi2001]. Um eine konsistente Repräsentation des Wissens bereitstellen zu können, muss der Weltmodellagent mindestens folgende Aufgaben lösen [Reichardt96]:

Matching	<i>Zuordnung</i> neuer Messungen zu bereits dem Weltmodell bekannten Objekten
Set & Forget	<i>Aufnehmen</i> und <i>Löschen</i> von Objekten
Merge & Split	<i>Zusammenfassen</i> und <i>Auseinanderbrechen</i> von Objekten
Filter & Extrapolation	<i>Glätten</i> der fusionierten Messdaten und <i>Weiterführung</i> von Objekten und Extrapolation derer Werte, falls keine Messdaten geliefert wurden

Weiterhin ist die *Prädiktion* der Zustände der Objekte bezüglich zukünftiger Zeitpunkte unter Abschätzung deren Verhaltens eine Aufgabe des Weltmodells, um ein "voraussehendes" Agieren des Fahrerassistenzsystems zu ermöglichen und Verkehrsszenen (z. B. mögliche Gefahrensituationen) durch den Situationsagenten auf Basis des Weltmodells zu erkennen. Dabei bestimmt der Situationsagent, welche Daten er vom Weltmodell benötigt. Der Weltmodellagent muss wiederum Verhandlungen mit dem Fusionsagenten führen, um das nötige Wissen zu erlangen (z. B. Änderung des Sensornetzwerkes, Änderung des Sehbereiches einer Kamera, etc.).

Im Rahmen dieser Arbeit stand der Entwurf von Datenstrukturen zur Speicherung des Wissens

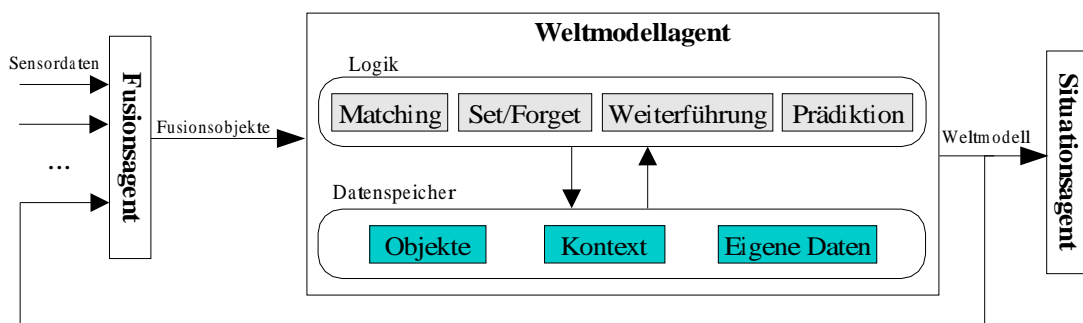


Abbildung 4.9: Aufgaben und Systemeinbettung des Weltmodellagenten

und zur Realisierung der Weiterführung und Prädiktion von Objekten im Vordergrund. Es wurden das oben beschriebene "Matching", "Set", "Forget", die Weiterführung ("Extrapolation") und die Prädiktion implementiert, um ein Beispiel für die Verwendung der eingeführten Klassen zu geben und einen funktionsfähigen Weltmodellagenten zum Test des Systems zur Verfügung zu haben.

Das “Matching“ wird teilweise bereits im Fusionsagenten vollzogen, indem die Messdaten den Fusionsvorgängen zugeordnet werden. Im Weltmodellagenten bedeutet das “Matching“ die Übergabe der aktuellen Beobachtungen in Form von Fusionsobjekten an die Bayes-Netze.

Die Abbildung 4.9 verdeutlicht die genannten Aufgaben des in dieser Arbeit realisierten Weltmodellagenten und dessen Einbettung in das Fahrerassistenzsystem. Die Trennung von Algorithmik und Daten dient in dem Bild nur der besseren Übersicht.

Dieses Kapitel beschreibt die Realisierung des Weltmodellagenten. Abschnitt 4.4.1 geht auf dessen Zyklus ein, in dem u. a. das “Matching“ und “Set & Forget“ durchgeführt wird. Danach wird die Datenstruktur des Weltmodells zur Speicherung der Eigen- und Umweltdaten vorgestellt. Anschließend wird im Abschnitt 4.4.3 die implementierte Logik zur Weiterführung und Prädiktion von Objekten in der Umwelt aufgezeigt.

4.4.1 Zyklus des Weltmodellagenten

Der Weltmodellagent ist ebenso wie der Fusionsagent (siehe Abschnitt 4.3) eine autonome Einheit, deren Zyklus endlos wiederholt von ANTS aufgerufen wird. Der Zyklus beginnt mit dem Einlesen der aktuellen Straßen- und Fahrzeugdaten, sowie der vom Fusionsagenten erzeugten Fusionsobjekte mit den Merkmalen der Objekte in der Umwelt. Diese Daten werden aus der globalen ANTS-Datenbank ausgelesen.

Für jedes in der Umwelt erkannte Objekt existiert im Weltmodell ein Netz, mit dessen Hilfe die Speicherung der Objektdaten aus der Vergangenheit und Gegenwart, sowie die Weiterführung und Prädiktion ermöglicht wird (mehr dazu im Abschnitt 4.4.3). Diese Netze werden mit der aktuell eingegangenen Menge der Fusionsobjekte verglichen. Dabei dient als Vergleichswert die eindeutige Identifikationsnummer, die vom Fusionsagenten für neu erkannte Objekte in der Umwelt vergeben wird. Existiert bereits ein Netz mit der Objekt-ID, wird das zugehörige Fusionsobjekt dem Netz als aktuelle Beobachtung übergeben. Dieser Vorgang realisiert das “Matching“. Wird kein Netz mit der Objekt-ID gefunden, legt der Weltmodellagent ein neues Netz an und ordnet ihm das entsprechende Fusionsobjekt als Beobachtung zu, was dem “Set“ entspricht. Netze, denen keine neuen Beobachtungen zugeordnet werden konnten, bleiben vorerst erhalten.

Somit wird die Fusionsobjektliste aus dem Fusionsagenten in das Weltmodell integriert, indem sie den einzelnen Netzen, die für die jeweiligen Objekte in der Umwelt stehen, zugeordnet wird. Der nächste Schritt im Zyklus ist das Update dieser Netze. Der erste Teil des Updates ist die Überprüfung, welche der Netze zu löschen sind. Der zweite Teil stößt bei jedem Netz die Berechnung der Werte an.

Das Löschen der Netze entspricht dem oben beschriebenen “Forget“. Ein Netz wird durch den in dieser Arbeit realisierten Weltmodellagenten gelöscht, wenn die letzte Detektion des zugehörigen Objekts in der Umwelt durch die Sensorik 60 s zurückliegt oder das Objekt sich außerhalb eines räumlichen Grenzbereiches befindet. Die dafür zugrundeliegende Forget-Logik kann beliebig erweitert oder angepasst werden.

Die übrigen Netze werden neu berechnet, wobei ihnen der aktuelle Zeitpunkt mitgeteilt wird. Mehr dazu ist im Abschnitt 4.4.3 nachzulesen. Nachdem die Berechnungen beendet sind, wird ein topologisches Netz zum aktuellen Zeitpunkt anhand der Berechnungen der oben genannten Netze erstellt, das eindeutig die momentan vom Fahrerassistenzsystem vermuteten Zustände der Objekte in der Umwelt beschreibt. Dieses topologische Netz wird in die globale ANTS-Datenbank geschrieben, wo es vom Fusionsagenten und (zukünftig) vom Situationsagenten gelesen werden

kann. Nähere Informationen zu dem topologischen Netz sind im Abschnitt 4.4.2.5 zu finden. Folgende Zusammenstellung beschreibt nochmal den Zyklus des Weltmodellagenten:

1. Einlesen der Straßen- und Fahrzeugdaten und der Fusionsobjektliste aus der globalen ANTS-Datenbank
2. Zuordnung der Fusionsobjekte zu den Berechnungsnetzen als aktuelle Beobachtung und gegebenenfalls neues Erstellen von Berechnungsnetzen für neu erkannte Objekte
3. Löschen der Berechnungsnetze anhand einer Forget-Logik
4. Update aller noch vorhandenen Berechnungsnetze
5. Erzeugen eines topologischen Netzes aus den Berechnungsnetzen heraus zum aktuellen Zeitpunkt
6. Schreiben des topologischen Netzes in die globale ANTS-Datenbank

Der Zyklus des Weltmodellagenten ist offen für Erweiterungen und Anpassungen. Das nächste Kapitel geht darauf ein, wie die Datenhaltung für die Umwelt- und Eigendaten entworfen wurde.

4.4.2 Datenstruktur des Weltmodells

Das in dieser Arbeit entworfene Weltmodell verwaltet die eigenen Daten, die Merkmale der Objekte in der Umwelt und den Kontext, wie es in [Levi2001] vorgeschlagen wird. Nach dem Paradigma der Objektorientierung wurden für die zu modellierenden Elemente der Welt Datenobjekte eingeführt, deren Strukturen durch Klassen beschrieben werden. Die Abbildung 4.10 zeigt die grobe Gesamtstruktur des Weltmodells in Form eines Klassendiagramms. Ein Objekt der Klasse **World-**

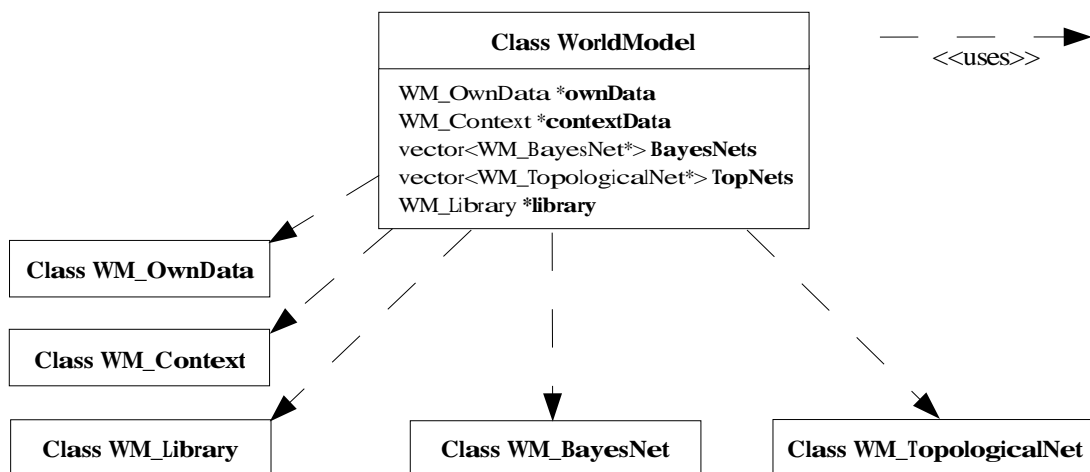


Abbildung 4.10: Klassendiagramm des Weltmodells

Model hält die Referenzen auf alle Teilbereiche des Weltmodells. Auf die eigenen Daten wird mit dem Attribut *ownData* verwiesen. Die Struktur der Klasse **WM_OwnData** ist im Abschnitt 4.4.2.2 nachzulesen. Auf den Kontext kann über das Attribut *contextData* vom Typ **WM_Context** zugegriffen werden (siehe Abschnitt 4.4.2.3). Die Klasse **WM_BayesNet** beschreibt die Struktur eines Bayes-Netzes als Grundlage für Berechnungen zur Weiterführung und Prädiktion und enthält die dafür notwendige Methodik. Ihr ist das separate Kapitel 4.4.3 gewidmet. Die Referenzen auf die Bayes-Netze werden in einer Liste gehalten (Attribut *BayesNets*), da die Anzahl der existierenden Bayes-Netze variabel ist. In der hier beschriebenen Realisierung des Weltmodells wird pro erkanntem Objekt in der Umwelt ein Bayes-Netz geführt.

Das topologische Netz (Klasse **WM_TopologicalNet**) wird in diesem Weltmodell als Schnittstelle nach außen verwendet, die eindeutig angibt, welche Objekte sich nach den Angaben der Sensoren und der Bayes-Netze zu einem bestimmten Zeitpunkt um das eigene Fahrzeug herum befinden. Damit es möglich ist, mehrere topologische Netze im Weltmodell speichern zu können (z. B. Netze der Vergangenheit, der Gegenwart und auch der Zukunft), werden die Referenzen auf die topologischen Netze in einer Liste (Attribut *TopNets*) gehalten. Nähere Informationen zur Realisierung der topologischen Netze sind im Abschnitt 4.4.2.5 gegeben. Zur Datenhaltung sonstiger Informationen (z. B. bekannte Transaktionen) wurde die Klasse **WM_Library** eingeführt (Abschnitt 4.4.2.6). Auf diese Bibliothek wird mit dem Attribut *library* verwiesen.

Folgender Abschnitt beschreibt die generische Struktur eines Merkmals, das im Weltmodell gespeichert wird.

4.4.2.1 Struktur eines Merkmals

In Systemen, die leicht erweiterbar und robust sein sollen, ist es notwendig, Informationen strukturiert zu speichern. Für ein Merkmal (z. B. Geschwindigkeit, Abstand, Farbe, etc.) gibt es neben dem Merkmalswert noch zusätzliche Informationen, die beschreiben, wann der Wert gesetzt wurde, welcher Sensor/Algorithmus den Wert ermittelt hat, wie zuverlässig der Wert ist, etc. Die Strukturierte Speicherung dieser Informationen zu jedem Merkmal ermöglicht die Klasse **WM_Attr**. Die Abbildung 4.11 zeigt deren in dieser Arbeit verwendete Struktur. Der Typ des

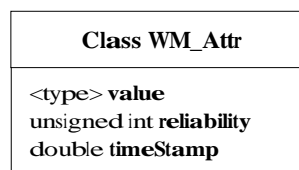


Abbildung 4.11: Attribute eines Merkmals

Wertes, der in *value* gespeichert wird, muss bei der Deklaration des Merkmals angegeben werden. Das Attribut *reliability* gibt Auskunft über die Zuverlässigkeit der Aussage, die mit dem Wert in *value* gemacht wird. Der Wertebereich ist von 0 bis 100, wobei 100 die höchste Zuverlässigkeit ausdrückt. In *timeStamp* steht der Zeitpunkt, zu dem der Wert gesetzt wurde.

WM_Attr ist beliebig erweiterbar, z. B. um Informationen bzgl. der Herkunft des Wertes oder einer textuellen Beschreibung [Schiek2001]. Beachtet werden muss jedoch, dass die Datenmen-

gen der Objekte klein bleiben sollten, um der Echtzeitanforderung durch geringe Übertragungs- und Rechenzeiten gerecht werden zu können. Aus diesem Grund wurde hier auf weitere Attribute verzichtet.

4.4.2.2 Eigene Daten

Die eigenen Daten beinhalten Informationen über das Fahrzeug, auf dem das Fahrerassistenzsystem eingesetzt wird und werden gemäß [Levi 2001] unterteilt in die eigenen Fahrzeugdaten, eigenen Erkennungsaufgaben, eigenen Fahrtabsichten, eigenen Steueraktionen und eigenen Transaktionen (siehe Kapitel 3.1.4). Die Abbildung 4.12 zeigt die Struktur der eigenen Daten. Dabei ist anzumerken, dass die eigenen Steueraktionen in der hier vorgestellten Implementierung nicht berücksichtigt wurden und deshalb nicht im Klassendiagramm erscheinen. Die Fahrzeugda-

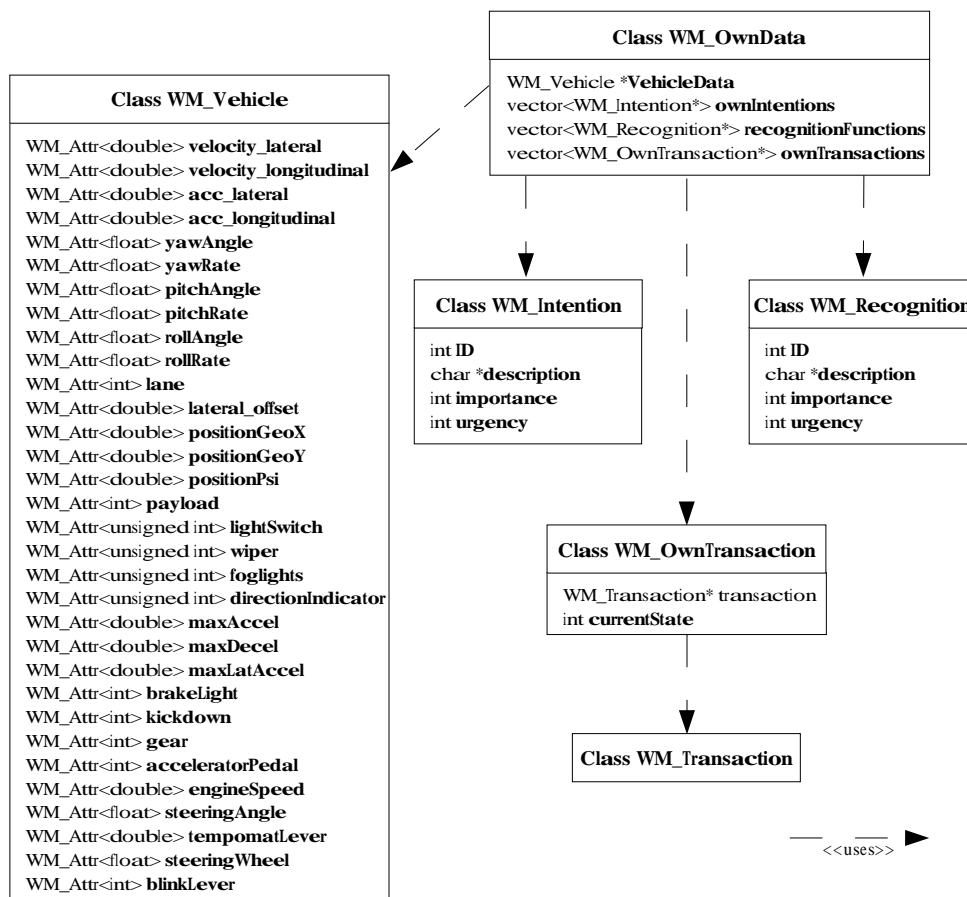


Abbildung 4.12: Klassendiagramm der eigenen Daten

ten werden durch die Klasse **WM_Vehicle** strukturiert und unter *VehicleData* gespeichert. Jedes der Attribute von **WM_Vehicle** ist mit der Klasse **WM_Attr**, die im Abschnitt 4.4.2.1 erklärt wird, deklariert. Nachfolgend werden die Attribute der Fahrzeugdaten genauer beschrieben:

velocity_lateral:	laterale Geschwindigkeit V_{Lat} in [$\frac{m}{s}$] <ul style="list-style-type: none"> • $V_{Lat} > 0 \Rightarrow$ Fahrzeug bewegt sich nach rechts • $V_{Lat} < 0 \Rightarrow$ Fahrzeug bewegt sich nach links
velocity_longitudinal:	longitudinale Geschwindigkeit V_{Long} in [$\frac{m}{s}$] <ul style="list-style-type: none"> • $V_{Long} > 0 \Rightarrow$ Fahrzeug bewegt sich vorwärts • $V_{Long} < 0 \Rightarrow$ Fahrzeug bewegt sich rückwärts
acc_lateral:	laterale Beschleunigung in [$\frac{m}{s^2}$]
acc_longitudinal:	longitudinale Beschleunigung in [$\frac{m}{s^2}$]
yawAngle:	Gierwinkel in [rad]
yawRate:	Gierwinkeländerungsrate in [$\frac{rad}{s}$]
pitchAngle:	Nickwinkel in [rad]
pitchRate:	Nickwinkeländerungsrate in [$\frac{rad}{s}$]
rollAngle:	Rollwinkel in [rad]
rollRate:	Rollwinkeländerungsrate in [$\frac{rad}{s}$]
lane:	Nummer der Spur auf der Strasse (siehe Abschnitt 4.4.2.3)
lateral_offset:	seitlicher Versatz von der Spurmitte in [m]
positionGeoX:	X Koordinate der GPS-Position in [$^\circ$]
positionGeoY:	y Koordinate der GPS-Position in [$^\circ$]
positionPsi:	Winkel zur Nord-Süd-Achse in [$^\circ$]
payload:	Zuladungsinformation
lightSwitch:	Status der Lichtanlage
wiper:	Scheibenwischerstatus
foglights:	Status der Nebelscheinwerferanlage
directionIndicator:	Fahrtrichtungsangabe
maxAccel:	maximale Längsbeschleunigung in [$\frac{m}{s^2}$]
maxDecel:	maximale Längsverzögerung in [$\frac{m}{s^2}$]
maxLatAccel:	maximale Querschleunigung in [$\frac{m}{s^2}$]
brakeLight:	Status des Bremslichtes
kickdown:	Information über einen evt. Kickdown
gear:	aktueller Gang
acceleratorPedal:	Status des Gaspedals
engineSpeed:	Motordrehzahl
steeringAngle:	Lenkwinkel in [$^\circ$]
tempomatLever:	eingestellte Wunschgeschwindigkeit
steeringWheel:	Lenkradwinkel in [$^\circ$]
blinkLever:	Status der Blinkanlage

Hinweis: Im Kapitel 3.2.2.3 stehen Informationen über die Akquirierung der oben aufgeführten Merkmale.

Die eigenen Fahrabsichten werden in der Liste *ownIntentions* verwaltet. Fahrabsichten sind in diesem Zusammenhang z. B. das Fahrziel oder ein Fahrmanöver [Levi2001]. Neben der eindeu-

tigen Identifikation der Absicht sind Platzhalter für die textuelle Beschreibung derselben und Variablen für die Wichtigkeit und die Dringlichkeit vorgesehen. Eine Fahrtabsicht wird durch die Klasse **WM_Intention** modelliert. Deren Attribute haben folgende Bedeutung:

ID	Identifikation einer Absicht
description	Beschreibung der Absicht
importance	Wichtigkeit der Absicht
urgency	Dringlichkeit der Absicht

Analog zu den Absichten wurden die Erkennungsaufgaben (Klasse **WM_Recognition**) entworfen. Beispiele für Erkennungsaufgaben sind die Detektion von Objekten, deren Identifikation oder die Erkennung der Fahrspur. Die Erkennungsaufgaben werden in der Liste *recognitionFunctions* verwaltet. In dieser Arbeit wurden die Absichten und Erkennungsaufgaben nicht implementiert, sondern lediglich Platzhalter dafür geschaffen.

Eine Transaktion beschreibt die Durchführung einer Aufgabe zum Erreichen eines Zieles. Eine Fahrtabsicht kann demnach durch eine Reihe von Transaktionen realisiert werden. Beispiele für Transaktionen sind das Überholen, Abbiegen und das Stoppen. Die eigenen Transaktionen sind den Transaktionen anderer Fahrzeuge gleich. Deshalb wurde für Transaktionen eine gemeinsame Klasse **WM_Transaction** eingeführt, deren Struktur unter 4.4.2.6 nachgelesen werden kann. Um den aktuellen Status innerhalb der durchgeführten eigenen Transaktion mitführen zu können, wurde die Klasse **WM_OwnTransaction** entworfen. Sie enthält folgende Attribute:

transaction	Verweis auf eine in der Bibliothek (Abschnitt 4.4.2.6) gespeicherten Transaktion
currentState	ID des aktuellen Status innerhalb der durchgeführten Transaktion

Die eigenen Transaktionen werden in der Liste *ownTransactions* verwaltet.

4.4.2.3 Kontext

Der Kontext enthält Informationen über die aktuellen Straßendaten, die Umweltbedingungen und die Verkehrszeichen. Des Weiteren wird ein Verweis auf eine Karte als Grundlage zur Planung des Weges zum Ziel, zum Erhalt von Straßenparametern und sonstigen Merkmalen der Umgebung vorgesehen, diese wurde jedoch in dieser Arbeit nicht verwendet. Die Struktur des Kontextes ist in der Abbildung 4.13 zu sehen. Die Klasse **WM_Context** hält die Referenzen auf die Teilbereiche des Kontextes:

streetData	Daten der Straße bzgl. der aktuellen Position des eigenen Fahrzeugs
mapData	Karte mit Informationen über die Straße, Orte, Umgebung, etc.
envData	Informationen über die Umweltbedingungen
trafficLaws	aktuell gültige Verkehrsordnung

Die Daten der Straße werden durch die Klassen **WM_Street** und **WM_Lane** strukturiert. Dabei beziehen sich die Angaben der Klasse **WM_Street** auf die gesamte Straße, die sich aus den einzelnen Spuren zusammensetzt (siehe auch Abschnitt 3.2.2.1):

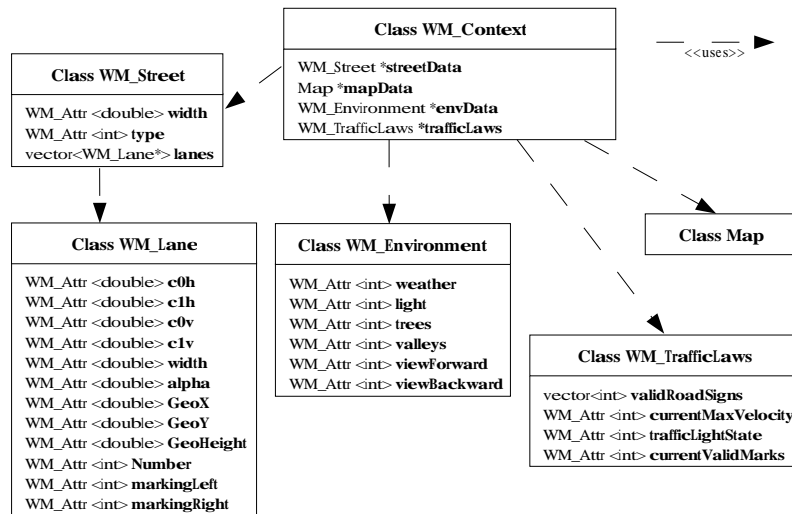


Abbildung 4.13: Klassendiagramm des Kontextes

- width** Gesamtbreite in $[m]$
type Klasse der Straße; kodiert wurden: “Unbekannt“, “Autobahn“, “Landstraße“, “Stadtstraße“ und “Feldweg“
lanes Liste aller bekannten Spuren, geordnet von links nach rechts

Die Gesamtbreite *width* ergibt sich aus den Breiten der einzelnen Spuren. Die Fahrbahnränder (Seitenstreifen) können gegebenenfalls auch als Spuren in die Liste *lanes* aufgenommen werden. Der Wert von *type* war aufgrund der Autobahnzenarios in dieser Realisierung konstant “Autobahn“. Die Attribute der Klasse **WM_Lane**, die eine einzelne Spur beschreibt, haben folgende Bedeutung:

- c0h** horizontale Krümmung in $[\frac{1}{m}]$
c1h horizontale Krümmungsänderung in $[\frac{1}{m}]$
c0v vertikale Krümmung in $[\frac{1}{m}]$
c1v vertikale Krümmungsänderung in $[\frac{1}{m}]$
width Breite in $[m]$
alpha Querneigung in $[rad]$
GeoX Geometrische Breite der Spurmitte in $[^\circ]$
GeoY Geometrische Länge der Spurmitte in $[^\circ]$
GeoHeight Höhe der Spurmitte über dem Meeresspiegel in $[m]$
number Nummer der Spur im Kontext aller benachbarten Spuren (Durchnummerierung aller Spuren von links nach rechts)
markingLeft linke Spurmarkierung; kodiert wurden: “Unmarkiert“, “Durchgezogen“ und “Gestrichelt“
markingRight rechte Spurmarkierung; Kodierung wie bei **markingLeft**

In dieser Arbeit wurden die Werte der Straße und der Spuren durch eine GPS-Schnittstelle direkt aus der Simulation geliefert (Abschnitt 4.2.3). Von welchen Sensoren die Merkmale einer Spur zusätzlich ermittelt werden können, wird im Abschnitt 3.2.2.2 beschrieben.

Die Klasse **Map** wurde in dieser Arbeit nicht verwendet bzw. implementiert, sie dient hier als Platzhalter für eine später zu integrierende Karte.

Den Zustand der Umwelt modelliert die Klasse **WM_Environment**:

weather	aktuelles Wetter; kodiert wurden: “niederschlagsfrei“, “regnen“ und “schneien“
light	aktuelle Lichtverhältnisse; kodiert wurden: “hell“, “halbdunkel“ und “dunkel“
trees	Bewaldung am Straßenrand; kodiert wurden: “keine“, “magere“ und “dichte“
valleys	Höhenunterschiede in der Umgebung; kodiert wurden: “keine“, “hügelig“ und “bergig“
viewForward	Sichtweite nach vorn in [<i>m</i>]
viewBackward	Sichtweite nach hinten in [<i>m</i>]

Die Werte für die Umgebungsmerkmale wurden in dieser Arbeit “per Hand“ gesetzt. Die Akquirierung dieser Merkmale durch die Sensoren (siehe Abschnitt 3.2.2.5) stellt eine zukünftige Aufgabe dar.

Auch das Wissen über die aktuell gültigen Verkehrszeichen ist nach [Levi 2001] ein Bestandteil des Kontextes. Die Klasse **WM_TrafficLaws** modelliert dieses Wissen:

validRoadSigns	Liste der gültigen Verkehrszeichen
currentMaxVelocity	aktuelle Höchstgeschwindigkeit in [$\frac{m}{s}$]
trafficLightState	Status der aktuellen Ampel; kodiert wurden (vereinfacht): “grün“, “gelb“, “rot und gelb“ und “rot“
currentValidMarks	gültige Markierung auf der aktuellen Spur; kodiert wurden: “linker Pfeil“, “rechter Pfeil“, “Pfeil geradeaus“ und die Kombinationen derselben

Als Verkehrszeichen wurden die Schilder “Zulässige Höchstgeschwindigkeit 100“ und “Ende sämtlicher Streckenverbote“ kodiert. Die Merkmale von *trafficLaws* wurden analog zu den Umgebungsdaten *envData* in dieser Arbeit manuell mit Werten gefüllt.

Die Attribute der beschriebenen Klassen, die direkt Aussagen über Zustände oder Eigenschaften machen, sind, wie im Klassendiagramm 4.13 zu sehen, vom Typ **WM_Attr** (siehe Abschnitt 4.4.2.1) und haben damit zusätzliche Informationen bzgl. der Zuverlässigkeit und dem Zeitpunkt der Eintragung.

4.4.2.4 Objekt in der Umwelt

Ein in der Umwelt erkanntes Objekt, das bereits alle Fusionsresultate enthält (in [Levi 2001] Entscheidungsobjekt genannt) und als solches den Fusionsagenten (siehe Abschnitt 4.3) verlässt, wird durch die Klasse **WM_Object** modelliert. Sie ist abgeleitet von der Klasse **FUSION_Fusionsobjekt** (siehe Abschnitt 3.3.2). Das Bild 4.14 zeigt das zugehörige Klassendiagramm.

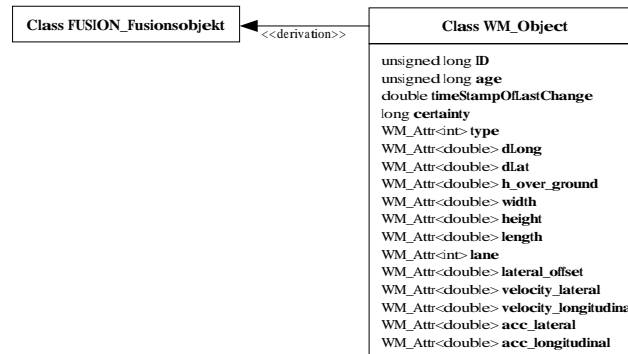


Abbildung 4.14: Attribute eines Objekts in der Umwelt

Die Objekte in der Umwelt werden direkt aus dem Fusionsprozess entnommen und dem Weltmodell übergeben. Diese Liste entspricht den Beobachtungen, welche die Eingabe für die Bayes-Netze (siehe Abschnitt 4.4.3) darstellen. Die Attribute der Klasse **WM_Object** haben folgende Bedeutung:

ID	eindeutige Identifikation eines Objekts (ganzzahlige positive Zahl)
age	Anzahl der Zyklen, in denen das Objekt erkannt wurde
timeStampOfLastChange	Zeitpunkt, zu dem das letzte mal ein Merkmal geändert wurde
certainty	Gewissheit über die Existenz des Objektes; Wertebereich ist 0 bis 100, wobei 100 die totale Gewissheit darstellt
type	Klassifizierung des Objektes; kodiert wurden “unbekannt“, “PKW“, “LKW“, “Motorrad“, “Fahrrad“, “Fußgänger“, “Ball“, “Verkehrszeichen“ und “Ampel“
dLong	longitudinaler Abstand relativ zum eigenen Fahrzeug in [m]
dLat	seitlicher Abstand relativ zum eigenen Fahrzeug in [m]
h_over_ground	Höhe des Objektreferenzpunktes in [m]
width	Breite des Objekts in [m]
height	Höhe des Objekts in [m]
length	Länge des Objekts in [m]
lane	Nummer der Spur, auf der sich Objekt befindet (siehe Abschnitt 4.4.2.3)
lateral_offset	seitlicher Versatz von der Spurmitte in [m]
velocity_lateral	laterale Relativgeschwindigkeit V_{Lat} in [$\frac{m}{s}$]
	<ul style="list-style-type: none"> • $V_{Lat} > 0 \Rightarrow$ Objekt bewegt sich relativ zum eigenen Fahrzeug nach rechts • $V_{Lat} < 0 \Rightarrow$ Objekt bewegt sich relativ zum eigenen Fahrzeug nach links

velocity_longitudinal:	longitudinale Relativgeschwindigkeit V_{Long} in $[\frac{m}{s}]$ <ul style="list-style-type: none"> • $V_{Long} > 0 \Rightarrow$ Objekt bewegt sich relativ zum eigenen Fahrzeug vorwärts • $V_{Long} < 0 \Rightarrow$ Objekt bewegt sich relativ zum eigenen Fahrzeug rückwärts
acc_lateral	laterale Relativbeschleunigung in $[\frac{m}{s^2}]$
acc_longitudinal	longitudinale Relativbeschleunigung in $[\frac{m}{s^2}]$

Der Wert von *ID* wird vom Fusionsagenten vergeben, sobald ein neues Objekt erkannt wird (siehe Abschnitt 4.3.3.1. Die relative Positionsangabe der Objekte und die zugehörigen Koordinatensysteme werden im Anhang B beschrieben.

4.4.2.5 Topologisches Netz

Durch die Klasse **WM_TopologicalNet** wird eine einfache egozentrische Karte mit topologischen Informationen beschrieben. Zugrunde liegt eine Einteilung der Umgebung des Versuchsträgers in 3 Teilbereiche, die durch die symbolischen Positionen **back**, **front** und **beside** benannt werden. Die Abbildung 4.15 zeigt das Klassendiagramm.

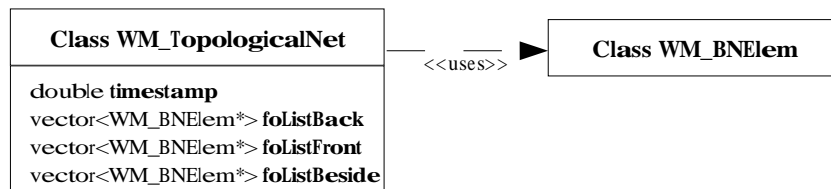


Abbildung 4.15: Klassendiagramm des Topologischen Netzes

Die Knoten des Netzes sind vom Typ **WM_BNElem** und entsprechen damit den Knoten von Bayes Netzen (Abschnitt 4.4.3). Dadurch sind neben den Informationen über die kinematischen Eigenschaften, die Position, die Ausmaße, etc. (siehe Abschnitt 4.4.2.4), die durch die Klasse **WM_Object** modelliert werden, zusätzlich die symbolische Position und eine anzunehmende Transaktion des Objektes in der Umwelt zugänglich.

Jede der Listen ist sortiert nach der longitudinalen Entfernung der Objekte. Dadurch wird ein systematischer und schneller Zugriff auf die Elemente des Netzes ermöglicht.

Das Einfügen von Objekten in das Topologische Netz wird anhand der metrischen Informationen über den longitudinalen Abstand und die Fahrzeuglänge vorgenommen. Wie die Koordinatensysteme und der Abstand in dieser Arbeit definiert wurden, ist im Anhang B nachzulesen.

Folgendes Regelsystem wird für das Einfügen eines Objektes in das Netz angewendet:

- E entspricht dem Beobachterfahrzeug
- F entspricht einem Objekt in der Umwelt von E
- F wird in die Liste foListBack eingefügt, wenn gilt:
 $DistLong(F) \leq (0 - Length(E))$
- F wird in die Liste foListFront eingefügt, wenn gilt:
 $(DistLong(F) - Length(F)) \geq 0$
- F wird in die Liste foListBeside eingefügt, wenn gilt:
 $(DistLong(F) > (0 - Length(E))) \wedge ((DistLong(F) - Length(F)) < 0)$

Das Topologische Netz wird aus den Bayes Netzen (Abschnitt 4.4.3) heraus aufgebaut. Der Vorgang wird im Zyklus des Weltmodellagenten nach der Berechnung der Bayes Netze angestossen. Dafür liegt der folgende Algorithmus zugrunde:

Aufrufparameter:

time Zeitpunkt, für den das Topologische Netz aufgebaut werden soll

Logik:

1. hole Referenz auf alle Bayes Netze
2. führe für jedes Bayes Netz durch:
 - (a) hole Zeitscheibe zum Zeitpunkt **time**
 - (b) merke die Knoten der Zeitscheibe vor, die den höchsten Wahrscheinlichkeitswert für den Zustand haben
 - (c) wähle den Knoten aus den vorgemerkten Knoten aus, dessen Wahrscheinlichkeitswert für die angegebene Transaktion am höchsten ist
 - (d) füge den gefundenen Knoten nach dem oben angegebenen Regelsystem in das Topologische Netz ein

Dadurch ist eine Verbindung vom Bayes Netz zum Topologischen Netz hergestellt. Gleichzeitig werden bei dem beschriebenen Vorgang die zuverlässigsten Informationen aus den Bayes Netzen herausgefiltert. Es wird entschieden, wie die Umwelt des Versuchsträgers zu einem Zeitpunkt angenommen wird. Darauf kann die Situationsinterpretation aufbauen. Allerdings wird immer nur eine mögliche Verkehrssituation ermittelt. Die in den Bayes-Netzen gehaltenen Alternativen, die ebenso für die Situationsinterpretation von Bedeutung sein können, um parallele mögliche Welten mitzuführen, müssten in Zukunft noch separat ausgelesen werden.

Das hier vorgestellte Topologische Netz lässt sich einfach erweitern. Die 3 genannten symbolischen Positionen, die in dieser Implementierung unterschieden werden, könnten verfeinert werden in genauere Positionsbeschreibungen (siehe Anhang A), für die jeweils eine separate Liste anzulegen ist.

4.4.2.6 Bibliothek

Es gibt Informationen, die dem Fahrerassistenzsystem bekannt sein müssen, die aber durch keine Sensoren geliefert werden. Diese Informationen sind z. B. Resultate von Lernverfahren oder manuelle Eingaben. Ein Beispiel dafür ist die allgemeine Verkehrsregel ‘rechts vor links’, die das Fahrerassistenzsystem kennen sollte, wenn es z. B. auf einer gleichberechtigten Kreuzung in der Stadt agieren soll. Zur Verwaltung solcher Daten wurde die Klasse **WM_Library** eingeführt. Die Abbildung 4.16 zeigt das Klassendiagramm der Bibliothek. In dieser Arbeit enthält die Bibliothek

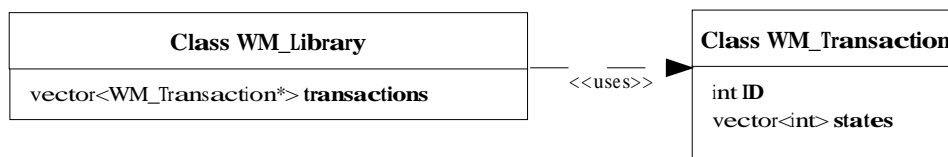


Abbildung 4.16: Klassendiagramm der Bibliothek

eine Liste von sogenannten Transaktionen, mit denen das Verhalten von Fahrzeugen beschrieben werden kann. Soll ein Beobachter in einem Fahrzeug, das überholt wird, kommentieren, wie sich das überholende Fahrzeug verhält, wird er möglicherweise sagen: „Das Fahrzeug ist links hinter mir, das Fahrzeug ist links neben mir und jetzt ist es links vor mir.“

Diese symbolische Beschreibung der Positionen anderer Objekte relativ zum Versuchsträger in der im Beispiel beschriebenen chronologischen Abfolge wird in diesem Zusammenhang als **Transaktion** bezeichnet. Eine Transaktion ist vergleichbar mit einem *Script* ([Schank77], [Reimer91]). Ein Script beschreibt die Sequenz von Teilereignissen, die erst beim Zutreffen einer Eingangsbedingung eintreten kann. Das Resultat der durch das Script dargestellten Ereignisfolge wird durch einen Ergebniszustand beschrieben. Auch die Teilereignisse können wieder durch Scripts beschrieben werden.

Durch die Transaktionen wird eine Verbesserung der Weiterführung und Prädiktion erreicht (siehe Abschnitt 4.4.3). Eine Transaktion wird durch die Klasse **WM_Transaction** modelliert. Sie besteht aus einer eindeutigen Nummer *ID* und einer Liste *states* von chronologisch geordneten Zuständen. Die Zustände entsprechen symbolischen Positionen (siehe Anhang A) bzw. Ereignissen beim Konzept der Scripts. Bei der Initialisierung der Bibliothek werden im Rahmen dieser Arbeit drei Transaktionen für verschiedene Überholmanöver angelegt.

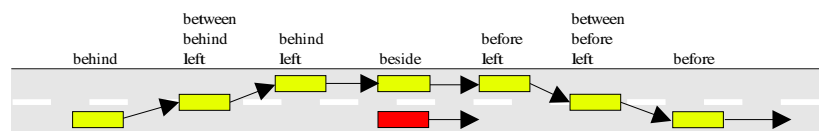


Abbildung 4.17: Transaktion für das Überholmanöver 1

Die Transaktion für das Überholmanöver 1 beschreibt die symbolischen Positionen eines überholenden Fahrzeugs, das zwei Spurwechsel (Aus- und Einscheren) vornimmt. Beim Überholvorgang 2 bleibt das überholende Fahrzeug auf der Überholspur:

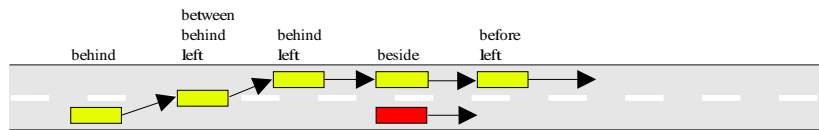


Abbildung 4.18: Transaktion für das Überholmanöver 2

Ein Überholvorgang ohne Spurwechsel wird durch das Überholmanöver 3 beschrieben.

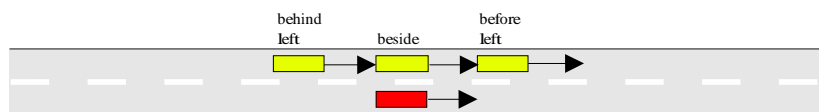


Abbildung 4.19: Transaktion für das Überholmanöver 3

Diese Arbeit beschränkt sich auf die dargestellten Transaktionen. Zukünftig bietet es sich an, mehr Transaktionen hinzuzufügen und eine Datenbank zu benutzen, in der die Transaktionen persistent gespeichert und verwaltet werden.

4.4.3 Realisierung der Weiterführung und Prädiktion von Objekten

Die Anforderung an das Weltmodell, die Eigen- und Umweltdaten des Fahrzeuges zu jedem Zeitpunkt konsistent zu repräsentieren, schließt ein, bekannte Objekte, die durch die Sensorik nicht erkannt werden konnten, intern weiterzuführen. Objekte können aus verschiedenen Gründen unerkannt bleiben. Z. B. können die Sensoren keine Messdaten zu einem Objekt liefern, wenn es sich

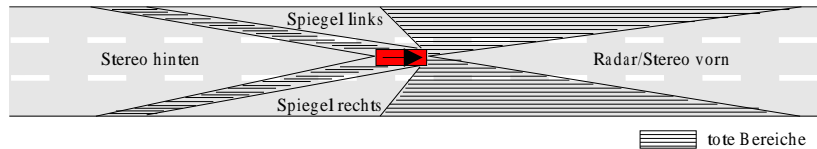


Abbildung 4.20: Bereiche, die nicht von der Sensorik abgedeckt werden

außerhalb des Sehbereiches, d. h. in einem “toten Bereich“ befindet, oder es von einem anderen Objekt verdeckt wird. Die “toten Bereiche“ um das in dieser Arbeit konfigurierte Versuchsfahrzeug skizziert die Abbildung 4.20. Durch Sensorausfälle können weitere Bereiche der Umwelt, in denen Objekte unerkannt bleiben, hinzukommen.

Die Prädiktion von Objektzuständen ist eine weitere Anforderung an das Weltmodell, durch die das Voraussehen und daraus abgeleitete Reaktionen (z. B. Warnungen oder Aktionen) des Fahrerassistenzsystems ermöglicht werden. Weiterhin ist man nicht abhängig vom Zeitpunkt der letzten Beobachtung, sondern kann die Zustände zu nachfolgenden Zeitpunkten berechnen. Dadurch ist es realisierbar, trotz des Zeitverlustes durch die Berechnungen im Fusions- und Weltmodellagenten dem Situationsagenten ein Weltmodell zur Realzeit zu übergeben.

Die Weltrepräsentation und die Realisierung der Weiterführung und Prädiktion sollte in die-

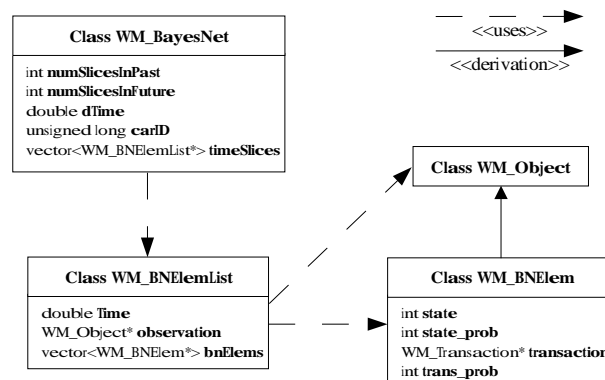


Abbildung 4.21: Klassendiagramm des Bayes-Netzes

ser Arbeit durch probabilistische Methoden erfolgen. Dafür standen verschiedene Ansätze zur Verfügung, u. a. dynamische Bayes-Netze, probabilistische temporale Netze und topologische Karten. Es wurde eine Kombination verschiedener Ansätze gewählt, die in der Studie [Levi2001]

vorgeschlagen wurde. Dabei wird ein dynamisches Bayes-Netz durch zusätzliche Relationen erweitert, wodurch eine topologische Karte entsteht. Im Kapitel 3.1.4 wurden bereits die Grundlagen der in dieser Arbeit implementierten Methodik zur Weiterführung und Prädiktion, die im Folgenden kurz "Bayes-Netz" genannt wird, gegeben.

Zur Realisierung des Bayes-Netzes wurden drei neue Klassen entworfen, die in der Abbildung 4.21 dargestellt werden. Durch die Klasse **WM_BNElem** wird die Datenstruktur eines Netzknotens, der den Zustand eines Objekts in der Umwelt zu einem Zeitpunkt repräsentiert, beschrieben. **WM_BNElem** ist von der Klasse **WM_Object** abgeleitet und enthält somit Attribute für detaillierte Positions-, Geschwindigkeits- und Größenangaben (siehe Kapitel 4.4.2.4). Erweiternd werden folgende Attribute hinzugefügt:

state	Information über die räumliche Position des Objektes in der Umwelt relativ zum Versuchsträger. Die Position ist symbolischer Art. Die Definition der symbolischen Positionen ist im Anhang A nachzulesen.
state_prob	Wahrscheinlichkeit für das Zutreffen der Information, die in state steht. In dieser Arbeit wurde dafür der Wertebereich 0 bis 100 definiert, wobei der Wert 100 die totale Gewissheit über das Zutreffen der Information darstellt.
transaction	Zeiger auf eine in der Bibliothek (siehe Abschnitt 4.4.2.6) gehaltene Transaktion, welcher das Objekt in der Umwelt folgen könnte.
trans_prob	Wahrscheinlichkeit dafür, dass die angegebene Transaktion wirklich durchgeführt wird (0..100).

Mit der topologischen Information, die durch den Wert im Attribut "state" gegeben ist, kann von der metergenaunen metrischen Position des Objektes in der Umwelt relativ zum Versuchsträger abstrahiert werden. Somit wird es möglich, das Verhalten anderer durch wenige Angaben zu beschreiben. Die Abfolge verschiedener symbolischer Positionen, die ein Fahrzeug bei einem Manöver (z. B. Spurwechsel, Überholmanöver, etc.) durchläuft, wurde im Abschnitt 4.4.2.6 als **Transaktion** eingeführt. Anhand einer Transaktion wird es leichter, die Zustände $S_i, i > n$ von einem gegebenen Zustand S_n aus zu berechnen. In Verknüpfung mit den Gesetzen der Kinematik erreicht man dadurch eine der Realität sehr nahe Weiterführung und Prädiktion. Wie diese Berechnung in der hier vorgestellten Implementierung durchgeführt wird, ist weiter unten nachzulesen. Es gibt Situationen, in denen man einem Objekt in der Umwelt mehr als eine Transaktion zuordnen kann. Zum Beispiel ist es möglich, dass ein überholendes Fahrzeug nach dem Überholvorgang auf seiner Spur bleibt, es könnte aber auch auf die Spur wechseln, auf der das überholte Fahrzeug fährt. Hier unterscheiden sich nur die Endzustände der Transaktionen. Um parallel mehrere Hypothesen über das Verhalten anderer Objekte mitführen zu können, wurde die Klasse **WM_BNElemList** eingeführt. Sie enthält folgende Attribute:

time	Zeitpunkt t, für den ein oder mehrere Zustände angegeben sind
observation	vom der Fusionsagenten erhaltene Beobachtung zum Zeitpunkt t
bnElems	Liste aller Zustände (Hypothesen) zum Zeitpunkt t, die Aussagen über ein Objekt in der Umwelt machen

Ein Objekt der Klasse **WM_BNElemList** wird im weiteren auch **Zeitscheibe** (time slice) genannt. Sobald eine neue Beobachtung zu einer Zeitscheibe geliefert wird, kann überprüft werden, inwieweit die bestehenden Hypothesen anhand von Transaktionen gerechtfertigt sind. Folgende Aktionen werden dabei angestoßen:

- Löschen der vorhandenen Hypothesen, die mit der Beobachtung konkurrieren
- Hinzufügen von Hypothesen, die sich aus der Beobachtung ableiten lassen, indem passende Transaktionen gesucht werden

Dafür wurde folgender Algorithmus verwendet:

1. Löschen aller vorhandenen Knoten der Zeitscheibe
2. Herausfiltern aller in der Bibliothek gespeicherten Transaktionen, welche die symbolische Position der Beobachtung als Zustand enthalten
3. von den gefundenen Transaktionen diejenigen auswählen, deren Folgezustand mit der momentanen Bewegung des beobachteten Objekts vereinbar ist
4. für jede der nun übrigen Transaktionen einen separaten Knoten (Instanz der Klasse **WM_BNElem**) erzeugen, einen Verweis auf die zugehörige Transaktion erstellen und die Daten der Beobachtung in die restlichen Felder eintragen; falls keine Transaktion gefunden wurde, einen einzelnen Knoten erzeugen und diesen mit den Beobachtungsdaten füllen

Die Verwaltung der Zeitscheiben und die Algorithmen zur Berechnung der Zustände wurden in der Klasse **WM_BayesNet** implementiert. Diese enthält folgende Attribute:

numSlicesInPast	Anzahl der Zeitscheiben, die in der Vergangenheit liegen und im Netz gespeichert bleiben sollen
numSlicesInFuture	Anzahl der Zeitscheiben, die in der Zukunft liegen und berechnet werden sollen
timeSlices	Liste der Referenzen auf die Zeitscheiben der Vergangenheit, Gegenwart und Zukunft
dTime	Zeitabstand zweier Zeitscheiben in ms
carID	eindeutige Identifikation des Objekts in der Umwelt, für das das Bayes Netz angelegt wurde

Die Gesamtanzahl der in einem Bayes Netz befindlichen Zeitscheiben TS wird bestimmt durch:

$$\#TS = numSlicesInPast + 1 + numSlicesInFuture$$

Die Liste der Zeitscheiben (Attribut *timeSlices*) ist geordnet nach den Zeitpunkten der Scheiben. Bei jeder Berechnung der Stati wird angegeben, welcher Zeitpunkt der Gegenwart entspricht. Daraufhin werden von der jüngsten Beobachtung an alle nachfolgenden Zeitscheiben sequenziell berechnet. Bevor der Algorithmus zum Update des Bayes-Netzes vorgestellt wird, folgt ein Beispiel (Abbildung 4.22), das ein Bayes-Netz mit topologischen Informationen für ein Objekt in der Umwelt anhand eines Überholmanövers zeigt. Zu sehen ist eine Verkehrsszene, in der das eigene Fahrzeug E mit einer Geschwindigkeit $V_E > 0$ auf einer zweispurigen Autobahn fährt. Von hinten nähert sich das Fahrzeug 1 auf der gleichen Spur mit einer Geschwindigkeit $V_1 > V_E$. In der Skizze sind zusätzlich 4 Zeitpunkte eingezeichnet und 2 anzunehmende Transaktionen des Fahrzeugs 1. Das zugehörige Bayes-Netz enthält in dem Beispiel nur Knoten für die 4 Zeitpunkte (BNElemList - Knoten). Jeder der Knoten enthält 2 BNElement-Knoten,

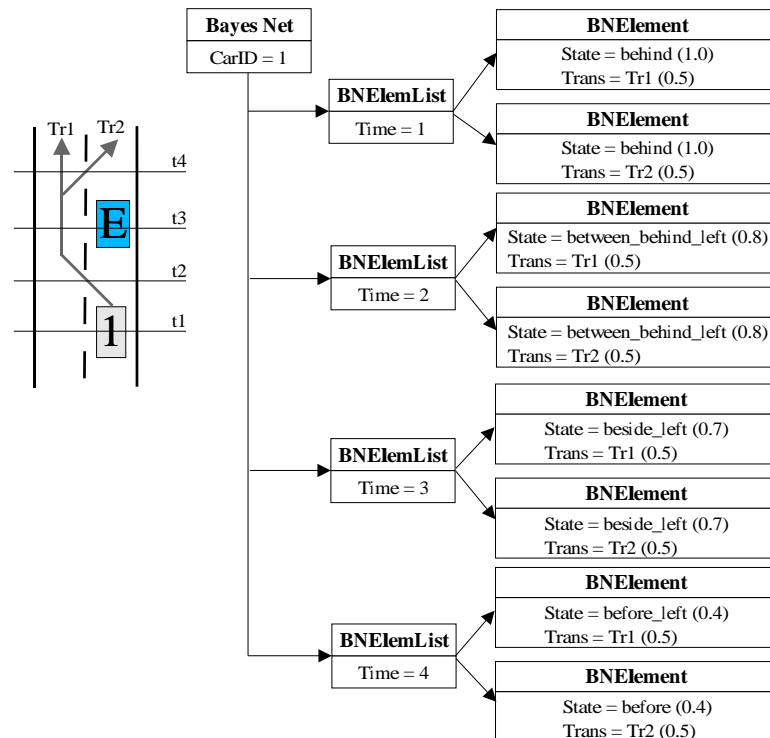


Abbildung 4.22: Beispiel eines Bayes Netzes mit topologischen Informationen

da man bereits vom Zeitpunkt $t1$ an auf 2 mögliche Transaktionen schliessen kann. Zusätzlich zu der Transaktion (Trans) sind jeweils die räumlichen symbolischen Positionen (State) angegeben. Die Werte in Klammern geben die Wahrscheinlichkeiten für das Zutreffen der Aussagen an.

Die Klasse **WM_BayesNet** besitzt eine öffentliche Methode *calculate*, mit der ein Update des Bayes Netzes angestoßen wird. Deren Aufruf wird im Zyklus des Weltmodellagenten vorgenommen, nachdem die neuesten Daten vom Fusionsagenten empfangen wurden. Nachfolgend wird der Algorithmus zum Update eines Bayes-Netzes vorgestellt.

Aufrufparameter:

currentTime	Zeitpunkt, der die Gegenwart datiert
ownData	Eigene Daten
context	Kontext
library	Bibliothek

Logik:

1. Verschieben aller Zeitscheiben $TS_j, j < currentTime$ an ihre neue Position
2. Löschen aller übrigen Zeitscheiben
3. Zuordnung der letzten Beobachtung $O_k, k \leq currentTime$, die vom Fusionsagenten kam, der ihr zugehörigen Zeitscheibe

4. Anhand der Beobachtung die möglichen Transaktionen und die symbolische Position für die Zeitscheibe TS_k ermitteln
5. Von der Zeitscheibe TS_k aus alle folgenden Zeitscheiben durch Prädiktion berechnen, falls eine Beobachtung O_k existiert, ansonsten die jüngste Zeitscheibe, die Knoten enthält, als Ausgangspunkt für die Prädiktion verwenden

Die Prädiktion des Zustandes S_n eines Fahrzeugs aus einem Zustand $S_m, m < n$ wird mit folgendem Algorithmus durchgeführt:

Aufrufparameter:

bnPrev	Zustand S_m
bnNew	Zu berechnender Zustand S_n (leeres Objekt)
ownData	Eigene Daten
context	Kontext
library	Bibliothek

Logik:

1. Ermitteln der Zeitspanne $deltaT$ zwischen den Zeitpunkten der Zustände S_m und S_n
2. Berechnen der neuen Position des Objekts mit Hilfe des zurückgelegten Weges in lateraler und longitudinaler Richtung durch Verwendung der Formel $s = V_m * deltaT$
3. Übernehmen weiterer unveränderlicher Attribute (Höhe, Breite, etc.) in den neuen Zustand
4. Berechnen der Spur und des Offsets des Fahrzeuges in der Spur anhand der neuen Position, den eigenen Fahrzeugdaten und dem Kontext
5. Ermitteln der symbolischen Position anhand der eigenen Fahrzeugdaten und dem Kontext
6. Berechnen der neuen Geschwindigkeit und Beschleunigung anhand der zugeordneten Transaktion

Durch diese Vorgehensweise wird die Trennung der Begriffe Weiterführung und Prädiktion von der methodischen Seite aus überflüssig. Die Weiterführung ist nur ein Sonderfall der Prädiktion, bei dem der Zeitpunkt des berechneten Zustandes in der Vergangenheit oder Gegenwart liegt.

In probabilistischen Netzen kann man durch Inferenzen aus Ereignissen bzw. Wirkungen Schlüsse auf Ursachen ziehen. Für ein dynamisches Bayes-Netz bedeutet dies, dass nicht nur vorwärtsgerichtet von Zuständen auf zeitlich nachfolgende Zustände geschlossen werden muss, wie es in dieser Arbeit gemacht wird, sondern auch die umgekehrte Richtung möglich ist. Ein Beispiel soll dies verdeutlichen. Wenn ein Fahrzeug hinter dem Versuchsträger über Funk mitteilt, dass es die nächste Autobahnabfahrt, die sich 100 m entfernt befindet, abfahren will, kann das Fahrerassistenzsystem mit hoher Wahrscheinlichkeit den zukünftigen Zeitpunkt voraussagen, an dem die Aktion durchgeführt wird. Aus dem sicheren Wissen über den Zustand des Fahrzeugs zu dem zukünftigen Abfahrtszeitpunkt kann durch Inferenz auf vorhergehende Zustände geschlossen werden. Z. B. kann ein Überholvorgang des Fahrzeugs nahezu ausgeschlossen werden,

da es bald von der Autobahn abfahren will. Derartige Inferenzen könnten zukünftig bei der Wahrscheinlichkeitsvergabe der einzelnen Zustände und Transaktionen mitwirken.

Kapitel 5

Test und Resultate

In diesem Kapitel werden die Tests, die mit dem in dieser Arbeit implementierten System gemacht wurden, und deren Ergebnisse beschrieben. Im Vordergrund steht dabei die Gewinnung der Erkenntnis, ob die in der Studie [Levi2001] vorgeschlagene flexible Strukturierung der Sensordatenfusion und die Methodik zur Weltrepräsentation im Kontext eines Fahrerassistenzsystems anwendbar sind. Die dabei hauptsächlich zu überprüfenden Punkte waren:

- Das Zusammenspiel der Fusions- und Weltmodellagenten
- Die Durchführbarkeit der dynamischen Konfiguration von Fusionsvorgängen im Fusionsagenten unter Verwendung der Kontrollobjekte und der Bereitstellung einer Methodenbibliothek
- Die komplementäre und konkurrierende Fusion der Sensordaten zur Sammlung und Übergabe aller verfügbaren Informationen an das Weltmodell
- Die Verwaltung und Aktualisierung des Weltmodells mit Hilfe einer topologischen Karte, die auf einem dynamischen Bayes-Netz beruht; besonderes Augenmerk wurde dabei auf die Konsistenz des Weltmodells und die Weiterführung von kurzzeitig von der Sensorik nicht detektierten Objekten gelegt

Zum Test des Systems diente eine Simulation als Sensordatenquelle. In ihr wurden mehrere Szenarien konfiguriert, von denen im folgenden Abschnitt drei vorgestellt werden. Danach wird auf die Testergebnisse eingegangen, wobei dem Fusionsagenten, Weltmodellagenten und schließlich dem Gesamtsystem separate Abschnitte gewidmet sind.

5.1 Testszenarien

Für die Szenarien wurde ein kurvenloser und ebener Straßenverlauf auf einer zweispurigen Autobahn gewählt. Die Szenarien sind Überholmanöver, durch die fast alle der oben genannten, zu überprüfenden Punkte getestet werden können. Die Szenarien wurden mit verschiedenen Sensornetzwerken ausgeführt, was verschiedene Ergebnisse, wie später zu lesen ist, hervorrief.

Szenario 1

Die erste Verkehrsszene besteht aus 2 Fahrzeugen. Das eine Fahrzeug ist der Versuchsträger (Ego), der mit konstanter Geschwindigkeit auf der rechten Spur fährt. Er beobachtet das Fahrzeug 1, das sich zuerst von hinten mit einer konstanten, aber größeren Geschwindigkeit auf der selben Spur nähert, danach einen Spurwechsel zur linken Spur durchführt, am Versuchsträger vorbeifährt und danach weiter auf seiner Spur bleibt. Die Abbildung 5.1 skizziert das beschriebene Szenario.



Abbildung 5.1: Szenario 1

Szenario 2

Die zweite Verkehrsszene ist ein wenig komplexer. In ihr ist zusätzlich zum überholenden Fahrzeug 1 aus dem ersten Szenario ein weiteres Fahrzeug 2 vor dem Versuchsträger auf der gleichen Spur plaziert, das langsamer als der Versuchsträger fährt. Nachdem das Fahrzeug 1 den Versuchsträger überholt hat, führt dieser selbst einen Spurwechsel auf die linke Spur aus, um eine Kollision mit dem Fahrzeug 2 zu vermeiden. Dieser Vorgang wurde in der Simulation fest hinterlegt, da in dem Fahrerassistenzsystem keine Aktorik implementiert wurde. Nachdem der

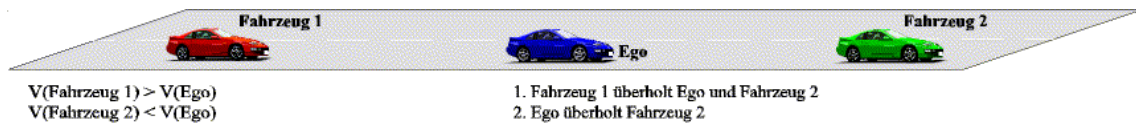


Abbildung 5.2: Szenario 2

Versuchsträger das Fahrzeug 2 überholt hat und dieses im Bereich der hinteren Stereokamera auftaucht, endet das Szenario. In der Abbildung 5.2 ist die Verkehrsszene nochmals dargestellt.

Szenario 3

Die dritte Verkehrsszene baut auf dem Szenario 2 auf. In ihr ist zusätzlich hinter dem überholenden Fahrzeug 1 aus dem zweiten Szenario ein weiteres Fahrzeug 3 mit der gleichen Geschwindigkeit, die das Fahrzeug 1 hat, auf der linken Spur plaziert. Das Fahrzeug 1 schert beim Spurwechsel vor dem Fahrzeug 3 auf der linken Spur ein. Danach fahren die Fahrzeuge 1 und 3 am Versuchsträger vorbei. Nun wechselt auch der Versuchsträger auf die linke Spur, damit er nicht mit dem Fahrzeug 2 kollidiert. Nachdem der Versuchsträger das Fahrzeug 2 überholt hat

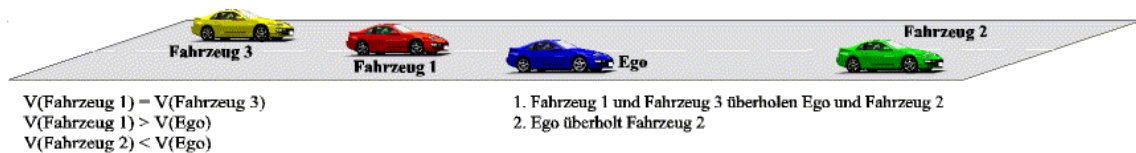


Abbildung 5.3: Szenario 3

und dieses im Bereich der hinteren Stereokamera auftaucht, endet das Szenario. Die geschilderte

Verkehrsszene ist in der Abbildung 5.3 dargestellt. Im nächsten Abschnitt wird der Test des Fusionsagenten beschrieben.

5.2 Fusionsagent

Die Aufgabe des Fusionsagenten im Rahmen dieser Arbeit ist hauptsächlich die Transformation und Verschmelzung der verschiedenen Objektbeschreibungen, die von den Sensoren kommen, zu einheitlichen Fusionsobjekten zur Weitergabe an den Weltmodellagenten (siehe Abschnitt 4.3). Dabei wird für jedes Objekt in der Umwelt ein separater Fusionsvorgang generiert, der speziell auf die Verarbeitung der Sensordaten des einen Objekts optimiert ist. Der in dieser Arbeit verwendete Fusionsvorgang ist im Abschnitt 4.3.2 erklärt.

Im Test des Fusionsagenten wurden folgende Aspekte untersucht:

- Dynamisches Generieren neuer Fusionsvorgänge für neu detektierte Objekte
- Zuordnung der Sensordaten zu existierenden Fusionsvorgängen
- Konzept der flexiblen Fusionsarchitektur
 - Sequentielles Durchlaufen der Fusionsebenen
 - Komplementäre und konkurrierende Sensordatenfusion
 - Algorithmenfusion
 - Verschmelzung der Resultate in Fusionsobjekten

Dazu wurden die im vorherigen Abschnitt vorgestellten Szenarien mit verschiedenen Sensornetzwerken durchgespielt. Der Test anhand des Szenarios 1 stand dabei im Vordergrund, da es die zu untersuchenden Aspekte bereits abdeckt. Nachfolgend werden einige Testdurchläufe beschrieben, wobei die verwendeten Sensoren, Fusionen und Ergebnisse genannt werden. Dabei werden nur die Sensoren angegeben, die zur Detektion anderer Objekte beitragen.

Testdurchlauf 1

In der Simulation wird das Szenario 1 ausgewählt. Beim eigenen Fahrzeug *Ego* sind die hintere und vordere Stereokamera und der Radarsensor in Betrieb.

Sobald das *Fahrzeug 1* in die Reichweite der hinteren Stereokamera kommt (100 m), liefert der Stereosensor einige Merkmale des *Fahrzeugs 1* (u. a. Position und Geschwindigkeit). Im Weltmodell ist zu diesem Zeitpunkt noch kein Fahrzeug bekannt. Deshalb wird für das *Fahrzeug 1* ein neuer Fusionsvorgang generiert und eine Identifikationsnummer vergeben. Die vom Stereosensor gelieferten Merkmale werden in den Eingabedatenslot "St (h)" des Fusionsvorgangs geschrieben. Im Slot "Intern" stehen die aktuellen eigenen Fahrzeugdaten, im Slot "Straße" die aktuellen Straßendaten. Die übrigen Eingabedatenslots bleiben leer. Nun wird der Fusionsvorgang gestartet. Die dabei stattfindenden Aktionen werden nachfolgend beschrieben:

1. Zusammenfassen der extrahierten Merkmale des *Fahrzeugs 1* in einem Fusionsobjekt
2. Spurzuordnung des detektierten Fahrzeugs anhand der eigenen Fahrzeugdaten und der Straße

3. Identitätsbestimmung des *Fahrzeugs 1* durch zwei verschiedene Algorithmen
4. Auswahl der zuverlässigsten Identitätsangabe und deren Eintrag in das Fusionsobjekt
5. Schreiben des Fusionsobjekts in den Ausgabedatenslot des Fusionsvorgangs

Das Zusammenfassen der Merkmale und die Berechnung der Spur sind der Merkmalsfusions-ebene zuzuordnen und entsprechen der komplementären Fusionsart. Die Vorgehensweise zur Identitätsbestimmung ist eine Algorithmenfusion, da auf einem Datensatz verschiedene Algorithmen angewandt werden und deren Ergebnisse fusioniert werden. Die Fusion der Resultate ist in diesem Fall konkurrierend.

Nachdem der Fusionsvorgang durchlaufen wurde, wird das Fusionsobjekt, das ein Modell für das *Fahrzeug 1* darstellt, dem Weltmodell übergeben.

Im nächsten Zyklus des Fusionsagenten werden erneut von der hinteren Stereokamera Merkmale zum *Fahrzeug 1* akquiriert. Der Fusionsagent hat Zugriff auf das Weltmodell des Fahrerassistenzsystems zum vorherigen Zeitpunkt. In diesem Weltmodell ist das Fusionsobjekt zum *Fahrzeug 1* bereits enthalten. Über einen räumlichen Positionsabgleich findet der Fusionsagent heraus, dass die von der Stereokamera gelieferte Beobachtung dem im Weltmodell gehaltenen Objekt entspricht. Daraufhin kann die Sensorbeobachtung in den Eingabedatenslot "St (h)" des Fusionsvorgangs, der im vorherigen Zyklus angelegt wurde, eingetragen werden. Nachfolgend wird wieder der Fusionsvorgang ausgeführt.

Die beschriebenen Aktionen wiederholen sich solange, bis das *Fahrzeug 1* auf die linke Spur wechselt und aus dem Sichtbereich der hinteren Stereokamera herausfährt. Nun wird es von keinem der Sensoren erkannt, wodurch die Eingabedatenslots des Fusionsvorgangs (außer "Intern" und "Straße") leer bleiben. Der Fusionsvorgang bleibt trotzdem erhalten, nur wird kein Fusionsobjekt an das Weltmodell geschickt. An dieser Stelle muss der Weltmodellagent dem Fahrerassistenzsystem weiterhelfen, indem er das *Fahrzeug 1* weiterführt und dessen Merkmalswerte abschätzt.

Nachdem das *Fahrzeug 1* im Sichtbereich der vorderen Stereokamera und des Radars auftaucht, liefern die beiden Sensoren Beobachtungen. Anhand des Weltmodells wird darauf geschlossen, dass die Beobachtungen zum *Fahrzeug 1* gehören. Deshalb werden die Eingabedatenslots "Radar" und "St (v)" des zugehörigen Fusionsvorgangs gefüllt. Der Fusionsvorgang liefert nun wieder ein Fusionsobjekt, das an den Weltmodellagenten weitergegeben werden kann.

Mit dem Testdurchlauf 1 wurden alle der oben genannten zu prüfenden Aspekte betrachtet. Es wurde erfolgreich ein neuer Fusionsvorgang für ein neu erkanntes Objekt konfiguriert und die Sensordaten wurden dem Fusionsvorgang als Eingabe zugeordnet. Im Fusionsvorgang wurden sowohl die Merkmals- als auch die Entscheidungsebene nacheinander durchlaufen, wobei komplementäre und konkurrierende Fusionen stattfanden. Ebenso wurde gezeigt, dass es möglich ist, mit der flexiblen Fusionsarchitektur Algorithmenfusionen durchzuführen. Auch das Ablegen der Daten in Fusionsobjekten funktionierte.

Testdurchlauf 2

Der zweite Testdurchlauf entspricht dem ersten, außer dass zusätzlich die beiden Spiegelkameras eingeschaltet werden. Dadurch wird der Sehbereich von *Ego* vergrößert. Am Fusionsvorgang muss nichts verändert werden, da die Eingabedatenslots für die Spiegelkameras bereits vorgesehen wurden. Sobald das *Fahrzeug 1* im Sehbereich der linken Spiegelkamera ist, werden dessen

Merkmale in den Eingabedatenslot “Sp (1)“ geschrieben.

Der zweite Testdurchlauf hat gezeigt, dass eine Veränderung des Sensornetzwerks möglich war, ohne dass ein Fusionsvorgang angepasst werden musste. Ebenso wird hier deutlich, dass der Fusionsagent die Sensorik vor den übrigen Ebenen im System verborgen halten kann.

Testdurchlauf 3

Die Sensorik ist dieselbe wie im Testdurchlauf 2. Es wurde jedoch das Szenario 2 durchgespielt, um zwei Fahrzeuge beobachten zu können. Nachdem beide Fahrzeuge erkannt wurden, existiert im Fusionsagenten für jedes ein Fusionsvorgang. Die Beobachtungen der Fahrzeuge werden in jedem Zyklus des Fusionsagenten ihren zugehörigen Fusionsvorgängen zugeordnet. Danach werden die Fusionsvorgänge unabhängig voneinander durchlaufen. Anschließend werden dem Weltmodell zwei Fusionsobjekte, die für die Fahrzeuge stehen, übergeben.

Durch den dritten Testdurchlauf konnte bestätigt werden, dass das parallele Existieren mehrerer Fusionsvorgänge und die Sensordatenzuordnung zu den Fusionsvorgängen in einer einfachen Szene korrekt verläuft.

Testdurchlauf 4

Im vierten Testdurchlauf wurde mit der gleichen Sensorik wie im vorherigen Testdurchlauf das Szenario 3 ausgeführt. Hier stand der Test der Zuordnung der Sensordaten zu den Fusionsvorgängen im Vordergrund, wenn mehrere Objekte sich räumlich sehr nahe kommen. Dies ist der Fall, wenn das *Fahrzeug 3* nach dem Überholvorgang im Zuge der Weiterführung in die Nähe des *Fahrzeugs 2* kommt. Dabei überlappen sich die Fahrzeuge sogar, da die Weiterführung des Weltmodells eine falsche Position des *Fahrzeugs 2* annimmt. Dadurch kam es zu falschen Zuordnungen der Daten zu den Fusionsvorgängen. Die in dieser Arbeit prototypisch implementierte Methode der Zuordnung der Messdaten zu den Fusionsvorgängen funktioniert demnach nur, wenn die Objekte in der Umwelt sich real annähernd so verhalten, wie es während der Weiterführung angenommen wird.

Die Testdurchläufe zeigen, dass die oben genannten zu prüfenden Aspekte durch den Fusionsagenten bei einfachen Szenarien weitgehend erfüllt werden. Die Fusionsvorgänge können dynamisch und statisch konfiguriert werden. Die verschiedenen Fusionsarten (komplementäre/konkurrierende Fusion, Algorithmenfusion) lassen sich in einem Fusionsvorgang unterbringen. Ein Fusionsvorgang kann in verschiedene Fusionsebenen unterteilt werden.

Die darüberliegende Logik, die anhand des Kontextes, des Sensornetzwerkes, der Situation, der Ziele, etc. selbständig die Konfiguration der Fusionsvorgänge vornimmt, muss auf dem in dieser Arbeit getesteten Grundkonzept aufsetzen. Die dynamische Konfiguration von Fusionsvorgängen ermöglicht die im Abschnitt 2.2.1 geforderte Eigenschaft der strukturellen Offenheit eines Agenten.

Der Mehraufwand bei der Entwicklung eines Systems, das eine solche flexible Fusionskomponente für die Sensordatenfusion enthält, wie sie in dieser Arbeit verwendet wurde, macht sich auf jeden Fall bezahlt, wenn einige der folgenden Aussagen zutreffen:

- Es liegt ein Sensornetzwerk zugrunde, das sich ändern kann
- Höhere Ebenen im System sollen von der Sensorik abstrahieren
- Es stehen mehrere Methoden zur Verarbeitung und Fusion der Sensordaten zur Verfügung,

die je nach Kontext und Zielen des Systems eingesetzt werden und deren Verwendung erst während der Laufzeit entschieden wird

- Das System soll selbständig die Konfiguration der Fusionsvorgänge vornehmen können
- Die Fusionen sollen nicht verstreut und unstrukturiert dort stattfinden, wo gerade ein Fusionsergebnis benötigt wird, sondern durch eine zentrale Fusionskomponente gesteuert werden mit dem gebündelten Wissen, welche Fusionen benutzt werden
- Es sollen mehrere Funktionen hintereinander ausgeführt werden, wobei die Ausgabedaten einer Funktion die Eingabedaten einer der nachfolgenden Funktionen sind
- Der Weg zu einem Fusionsresultat soll nachvollziehbar sein, wobei auch der Zugriff auf die Zwischenergebnisse erwünscht ist
- Es sollen mehrere Fusionsvorgänge parallel existieren und ausgeführt werden können

Bei zukünftigen Fahrerassistenzsystemen wird die Anforderung, Sensordaten flexibel fusionieren zu können, zunehmen. Es werden große Sehbereiche um das Fahrzeug herum benötigt und die zu erlangenden Informationen müssen zuverlässig sein. Je nach zu erlangender Information und Kontext werden verschiedene Fusionstechniken anzuwenden sein. Daher wird es unumgänglich sein, in Zukunft ein solches Konzept zur flexiblen Konfiguration von Fusionsvorgängen, wie es in dieser Arbeit geprüft wurde, einzuführen.

5.3 Weltmodellagent

Die Aufgabe des Weltmodellagenten im Rahmen dieser Arbeit ist die konsistente Repräsentation der Umwelt- und der Eigendaten. Dafür steht ihm neben der im Abschnitt 4.4.2 ausführlich betrachteten Klassenbibliothek für die Datenhaltung eine Logik zur Verwaltung der Objekte in der Umwelt zur Verfügung, um deren Test es in diesem Abschnitt geht.

Die von der Sensorik erkannten Objekte kommen in Form von Fusionsobjekten vom Fusionsagenten zum Weltmodellagenten. Diese werden, wie es im Kapitel 4.4 beschrieben wird, in das Weltmodell integriert, indem ein "Matching", "Set" und "Forget" stattfindet. Darüberhinaus gibt es eine Weiterführungslogik, damit die kurzzeitig von der Sensorik "aus den Augen" verlorenen Objekte nicht in Vergessenheit geraten. Zusätzlich wurde eine Logik zur Prädiktion realisiert, um zukünftige Objektzustände abschätzen zu können.

Basis für die Logik des Weltmodellagenten ist das im Abschnitt 4.4.3 vorgestellte probabilistische, um topologische Informationen erweiterte Netz in Kombination mit Transaktionen, die Vermutungen über das Verhalten von Verkehrsteilnehmern modellieren. Dabei wird für jedes Objekt in der Umwelt ein separates Netz gehalten. Die Zuordnung neuer Beobachtungen zu einem Netz ("Matching"), das Erzeugen von Netzen für neu erkannte Objekte ("Set") und das Löschen von Netzen für nicht mehr relevante Objekte ("Forget") wurde bei allen Szenarien korrekt durchgeführt. Im Folgenden wird hauptsächlich die Weiterführung und Prädiktion betrachtet, da mit ihnen die Brauchbarkeit der Methodik, die dem dynamischen Bayes-Netz zugrunde liegt, gezeigt werden kann.

Zur Visualisierung der Verkehrsszene während der Programmausführung wurde eine grafische Darstellung entwickelt, die sowohl die Straße, als auch die darauf befindlichen Fahrzeuge aus

der Vogelperspektive darstellt. Darüberhinaus besteht die Möglichkeit, die Zeitscheiben des Berechnungsnetzes einzuzichnen. Das Versuchsfahrzeug dient als fester Bezugspunkt, um den herum sich alles bewegt. Es sind auch die Bereiche eingezeichnet, die durch die Sensorik abgedeckt werden. Somit ist ersichtlich, wann ein Fahrzeug durch den Fusionsagenten erkannt wird und wann es nur durch die Logik der Weiterführung im Weltmodell gehalten wird. Die grafische Darstellung erleichterte sehr den Test des entwickelten Systems.

Ein Berechnungsnetz besteht aus mehreren Zeitscheiben, die einen bestimmten Zeitabstand untereinander haben. Für den Test wurden verschiedene Werte für die Anzahl der Zeitscheiben und den Zeitabstand gewählt. Je mehr Zeitscheiben existieren, desto länger dauert die Berechnung eines Netzes, die in jedem Zyklus des Weltmodellagenten vorgenommen wird. Wenn der Zeitabstand sehr groß gewählt wird, dann kann man mit dem Netz weit vorausschauen, die Berechnung wird aber sehr grob und ungenau. Bei sehr kleinen Zeitabständen ist die Prädiktion sehr genau, aber man kann nicht sehr weit vorausschauen. Hier musste ein Mittelweg gefunden werden, je nach dem, ob die Weiterführung oder die Prädiktion im Vordergrund steht und was die aktuellen Ziele und Aufgaben des Systems sind. Bei freier Fahrt auf der Autobahn z. B. kann ein größerer Zeitabstand angebracht sein, bei der Aufgabe eines Spurwechsels im dichten Verkehr ein kleinerer. Die Abbildung 5.4 zeigt einen Ausschnitt des Szenarios 1, in dem bei gleicher Zeitscheibenanzahl drei verschiedene Zeitabstände gewählt wurden. Dabei wurden jeweils die Zeitscheibe der Gegenwart und 20 Zeitscheiben der Zukunft eingezeichnet. Der Zeitabstand zwischen zwei Zeitscheiben ist bei a) 8 ms, b) 50 ms und bei c) 130 ms. Die Zeitscheibe der

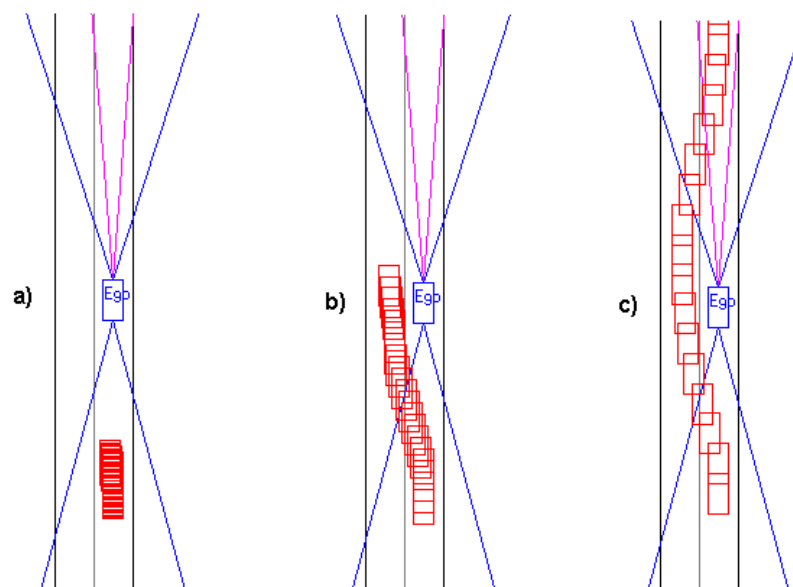


Abbildung 5.4: Drei dynamische Bayes-Netze mit unterschiedlichen Zeitabständen

Gegenwart ist bei jeder der Skizzen die unterste. Sehr schön zu sehen ist hier die Prädiktion unter Berücksichtigung einer möglichen Transaktion des Fahrzeugs 1, die mit einer bestimmten Wahrscheinlichkeit versehen ist. In dem abgebildeten Testdurchlauf stand für die Berechnung nur eine Transaktion zur Verfügung, deren Verlauf in der Skizze c) sehr gut sichtbar ist.

Soll das Berechnungsnetz nur zur Weiterführung dienen, reicht es, wenn es Zeitscheiben für die Vergangenheit und Gegenwart enthält. Zeitscheiben der Zukunft sind für die Weiterführung nicht

relevant. Für eine stabile Weiterführung haben sich im Test ein Zeitabstand von 4 ms und die Anzahl von 20 Zeitscheiben für die Vergangenheit bewährt. Bei anderen Konfigurationen kam es vor, dass bei einem Update des Netzes zum Zeitpunkt t keine Zeitscheiben des Netzes zum Zeitpunkt $t-1$ als Berechnungsgrundlage zur Verfügung standen, d. h. das Netz war leer und das Objekt in der Umwelt konnte nicht mehr weitergeführt werden. Wie die Konfiguration eines solchen Berechnungsnetzes im realen Einsatz bzgl. Zeitabstand und Anzahl der Zeitscheiben zu gestalten ist, muss in weiteren Arbeiten erst herausgefunden werden.

Die Prädiktion unter der Berücksichtigung mehrerer möglicher Transaktionen war ein weiterer zu testender Punkt. Hierzu diente auch wieder das Szenario 1 als Verkehrsszene. Dem Berechnungsnetz standen zwei verschiedene Transaktionen zur Verfügung. Die Abbildung 5.5 zeigt die zwei berechneten Möglichkeiten, die für das Verhalten des Fahrzeugs 1, nachdem es am eigenen Fahrzeug (Ego) vorbeigefahren ist, angenommen wurden. Die Zeitscheibe der Gegenwart

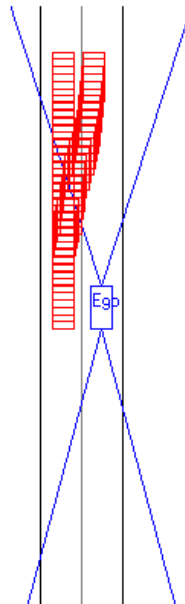


Abbildung 5.5: Prädiktion unter Berücksichtigung zwei möglicher Transaktionen

befindet sich auf der gleichen Höhe wie das eigene Fahrzeug (Ego). Alle weiteren zu sehenden Zeitscheiben sind mit zukünftigen Zeitpunkten datiert. Die eine Hypothese sagt aus, dass das überholende Fahrzeug wieder einen Spurwechsel zur rechten Spur durchführen wird. Die andere Hypothese behauptet, das Fahrzeug bleibt auf der linken Spur. Beide Hypothesen haben eine bestimmte Wahrscheinlichkeit. Aufgrund des Rechtsfahrgebotes wird der Spurwechsel zur rechten Spur wahrscheinlicher sein, solange sich kein weiteres Fahrzeug in der Nähe befindet. Dementsprechend werden auch die Wahrscheinlichkeiten für die einzelnen Positionen, die durch die Prädiktion berechnet werden, von der zugrundeliegenden Transaktion beeinflusst.

Interessant ist nun, wie sehr sich die prädierten Zustände der Objekte von den später wirklich beobachteten Zuständen unterscheiden. Dazu wurden zwei Testdurchläufe des Szenarios 1 mit unterschiedlicher Sensorausstattung durchgeführt. Einmal waren alle Sensoren angeschaltet, beim zweiten Durchlauf standen keine Spiegelkameras zur Verfügung, wodurch ein "toter Bereich" hinzukam. Die Abweichung der Werte in dem "toten Bereich" im Vergleich zur Beobachtung mit verfügbarer Spiegelkamera ist in der Abbildung 5.6 zu sehen. Hierbei sind nur die Zeitscheiben

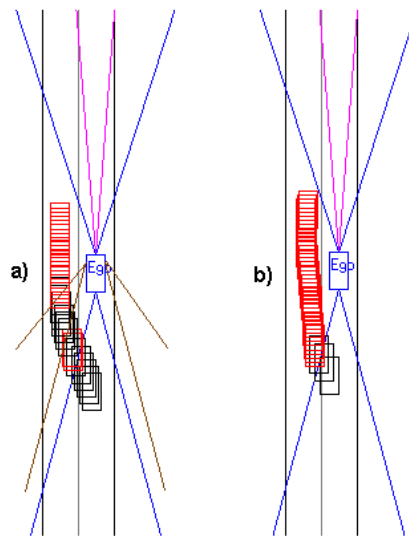


Abbildung 5.6: Vergleich von Schätzung und Beobachtung im Sehbereich der linken Spiegelkamera (in der Skizze a) ist die linke Spiegelkamera angeschaltet, in der Skizze b) müssen die Werte innerhalb des fehlenden Sehbereichs geschätzt werden)

der Vergangenheit und Gegenwart eingezeichnet. Die Zeitscheibe der Gegenwart entspricht dem jeweils obersten Rechteck. Es ist deutlich zu erkennen, dass in der Skizze a) das Fahrzeug eher die Mitte der linken Spur erreicht als das Fahrzeug in der Skizze b).

Befinden sich mehrere Fahrzeuge in der Umwelt des Versuchsträgers, wie es in den Szenarien 2 und 3 zu sehen ist, kann man für jedes einzelne Fahrzeug dieselben Ergebnisse beobachten, wie sie anhand des Szenarios 1 gezeigt wurden. Allerdings fehlt noch die Logik, die solche Transaktionen durch Vergabe einer geringen Wahrscheinlichkeit ausschließt, deren Durchführung z. B. zu Kollisionen führen würden. Ein Beispiel dafür ist das Szenario 2, in dem das Fahrzeug 1 nach dem Überholvorgang prinzipiell nicht auf die rechte Fahrspur wechseln würde, da sich dort das Fahrzeug 2 befindet. Im Abschnitt 6.2.2 werden einige der Punkte, die noch der Logik des Weltmodellagenten hinzugefügt werden sollten, genannt.

Die in [Levi2001] vorgeschlagene und in dieser Arbeit implementierte Methodik zur Weiterführung und Prädiktion, die auf dynamischen Bayes-Netzen mit topologischen Informationen beruht, hat sich in den Tests des in dieser Arbeit entwickelten Systems als sehr sinnvoll und brauchbar erwiesen. Die Berechnungen waren von linearem Aufwand, wobei allerdings nur die Beziehungen der Fahrzeuge relativ zum Versuchsträger und nicht untereinander berücksichtigt wurden. Besonders durch die Berechnungen anhand von Transaktionen konnte eine sehr realitätsnahe Weiterführung und Prädiktion beobachtet werden. Auch die entworfene Struktur des dynamischen Bayes-Netzes, mit der beliebig viele Hypothesen über mögliche Transaktionen verwaltet werden können, hat sich in den Tests bewährt. Die Brauchbarkeit der Methode unter Berücksichtigung der Beziehungen zwischen den verschiedenen erkannten Objekten und im Einsatz in einem realen Versuchsträger muss jedoch zukünftig noch untersucht werden.

Eine weitere Aufgabe ist das Hinterlegen zusätzlicher Transaktionen, die das mögliche Verhalten

der Fahrzeuge modellieren. Z. B. könnte das Fahrzeug 1 in der Abbildung 5.6, während es im “toten Bereich“ ist, abbremsen. Der hier getestete Weltmodellagent würde das Objekt mit gleicher Geschwindigkeit weiterführen, da er die Transaktion “Abbremsen“ nicht kennt, bis in den Sehbereich der vorderen Kamera hinein.

In jedem Zyklus des Weltmodellagenten wird aus den berechneten dynamischen Bayes-Netzen eine einfache topologische Karte zum aktuellen Zeitpunkt aufgebaut, wobei für jedes Fahrzeug der wahrscheinlichste Objektzustand in der Karte hinterlegt wird. Sie ist in dieser Arbeit die Grundlage für den Fusionsagenten, der überprüft, welche Objekte in der Umwelt dem System schon bekannt sind. Ein Nachteil der topologischen Karte ist, dass sie keine Informationen über die alternativen Zustände des Weltmodells enthält, die im dynamischen Bayes-Netz mitgeführt werden.

5.4 Gesamtsystem

Der Einsatz einer Multiagentenarchitektur und die damit verbundene Trennung der Aufgabenbereiche Sensordatenakquirierung, Sensordatenfusion, Weltmodellierung, Situationsinterpretation, etc. für Fahrerassistenzsysteme ist eine neue Vorgehensweise, deren Umsetzung im Rahmen dieser Arbeit begonnen wurde. Welche Vorteile und Probleme einer solchen Architektur sich bei der Realisierung und beim Test des Systems herausstellten, wird in diesem Abschnitt erläutert.

Das System umfasst in dieser Arbeit die Simulation, die Sensoren, die Kommunikation zwischen den Prozessen, ANTS, sowie den Fusions- und Weltmodellagenten. Die Modularisierung eines Systems in einzelne Komponenten und die Verwendung einer Softwarearchitektur für Multiagentensysteme (wie z. B. ANTS) bringen viele Vorteile mit sich:

- Übersichtlichkeit der Systemstruktur und der zu erledigenden Aufgaben
- verteilte Logik erleichtert die Verteilung der Prozesse auf verschiedene Rechner (Verteiltes System)
- Austauschbarkeit und Änderungsaufwand von Systemkomponenten werden verbessert
- der Ausfall einzelner Komponenten führt nicht unbedingt zum Ausfall des Gesamtsystems (Robustheit)
- einfache Systemkonfiguration durch Hinzufügen oder Entfernen von Komponenten
- Möglichkeit der Parallelisierung von Abläufen

Die Übersichtlichkeit der Aufgabenverteilung im System ist durch die agentenbasierte Architektur gegeben. Die Funktionseinheiten liefern die Sensordaten. Der Fusionsagent ordnet diese Fusionsvorgängen zu und führt die Fusionen durch. Der Weltmodellagent repräsentiert die Daten in einer konsistenten Struktur. Jede der Komponenten kann verbessert bzw. erweitert werden, ohne dass andere Komponenten davon betroffen sind. Zu beachten ist dabei, dass die Kommunikation zwischen den Komponenten korrekt bleibt.

Die Kommunikation zwischen den Sensorschnittstellen und dem Fusions- und Weltmodellagenten basierte im Rahmen dieser Arbeit nur auf dem Datentransfer von Sensor- und Weltmodelldaten. Die Architektur des Systems macht es leicht, weitere Kommunikationswege zu erstellen, über die

z. B. Aufträge bzw. Verhandlungen der Agenten ausgetauscht werden könnten.

Fallen Sensoren oder der Fusionsagent aus, können trotzdem über die Weiterführung des Weltmodellagenten über eine kurze Zeit die aktuellen Informationen bzgl. der Verkehrsszene aufrecht erhalten werden. Dieser Fall wurde explizit getestet. Das Sensornetzwerk bestand aus der vorderen und hinteren Stereokamera. Die Verkehrsszene entsprach dem Szenario 1. Das Fahrzeug 1 wurde von der hinteren Stereokamera in mehreren Zyklen detektiert. Während das Fahrzeug 1 sich noch hinter dem Versuchsträger auf der rechten Spur befand, wurde die hintere Stereokamera ausgeschaltet. Dies hatte zur Folge, dass das Fusionskontrollobjekt für das Fahrzeug 1 im Fusionsagenten keine neue Beobachtung bekam. Demzufolge wurde dem Weltmodellagenten für das Fahrzeug 1 kein Fusionsobjekt gesandt. Das Bayes-Netz zum Fahrzeug 1 setzte deshalb seine Berechnungen auf den alten Beobachtungen und Prädiktionen auf. Anhand der vermuteten Transaktion des Fahrzeugs 1 (Überholmanöver) konnte das Fahrzeug bis in den Sehbereich der vorderen Kamera weitergeführt werden. Allerdings konnten ab dem Ausschalten der hinteren Kamera keine neuen Objekte im hinteren Fahrzeugbereich erkannt werden.

Die Parallelisierung der Agenten verwaltet ANTS. Die Zyklen der Agenten werden so oft aufgerufen, wie es möglich ist. Um Rechenzeit zu sparen, wurde in den Fusions- und Weltmodellagenten eine Logik eingebaut, die den Zyklus des jeweiligen Agenten abbricht, falls keine neuen Eingabedaten verfügbar sind. Auf diese Weise konnte das System in seiner Ausführung beschleunigt werden.

Nachteile des Systems, die sich beim Test herausstellten, betreffen hauptsächlich den Datentransfer. Durch die verwendete Systemarchitektur ergaben sich:

- ein erhöhter Kommunikationsaufwand zwischen den Prozessen durch Verhandlungen und Datenaustausch
- eine lange Übertragungsdauer großer Datenmengen (z. B. Kamerabilder) über das Netzwerk

Aus diesen Gründen ist das System noch weit entfernt von der Echtzeitfähigkeit. Die Integration von Fusionsmethoden in den Fusionsagenten verlangt, dass die dafür benötigten Sensordaten über das Netz dem Fusionsagenten zur Verfügung gestellt werden müssen. Erst mit ausreichend schnellen Netzwerken wird dies möglich sein. Bis dahin müssen Kompromisse geschlossen werden, wie z. B. die Verarbeitung von Kamerabildern auf dem Rechner, auf dem sie vom Framegrabber empfangen wurden.

In dieser Arbeit haben sich innerhalb des Simulationssystems die Ideen, die in [Levi2001] bzgl. Systemarchitektur, Sensordatenfusion und Weltmodellierung eingeführt wurden, als machbar und brauchbar erwiesen. Der Einsatz und die Umsetzung der Ideen in einem realen Fahrzeug muss dieser Arbeit folgen, um deren Anwendbarkeit in Fahrerassistenzsystemen weiter zu erforschen.

Kapitel 6

Zusammenfassung und Ausblick

6.1 Zusammenfassung

An Fahrerassistenzsysteme werden immer höhere Anforderungen gestellt. Das Wissen über die Eigen- und Umweltdaten des Fahrzeugs gewinnt dabei an Bedeutung. Dazu ist die Erfassung von Signalen aus der Umwelt durch Sensoren, die Fusion und Weiterverarbeitung dieser Signale und die Modellierung der Eigen- und Umweltdaten in einer geeigneten Repräsentationsstruktur notwendig.

In dieser Arbeit wurde ein System prototypisch realisiert, das die Verarbeitung der Sensordaten bis hin zur Weltmodellierung anhand neuartiger Konzepte vornimmt. Eines der neuartigen Konzepte umfasst die Architektur des Systems. In [Levi2000] wird empfohlen, Fahrerassistenzsysteme nach dem Paradigma von Multiagentensystemen zu entwerfen. Darum wurde das in dieser Arbeit realisierte System unter Verwendung von ANTS in verschiedene Komponenten gegliedert, die für die Sensordatenbeschaffung (Sensorschnittstellen), Sensordatenfusion (Fusionsagent) und für die Weltmodellierung (Weltmodellagent) verantwortlich sind. Damit findet eine Trennung von Sensordatenbeschaffung, Sensordatenverarbeitung und Weltmodellierung statt.

Innerhalb des Fusionsagenten wurde eine flexible Fusionsarchitektur, mit der Fusionsvorgänge statisch und dynamisch konfiguriert werden können, verwendet und getestet. Dieser Fusionsarchitektur liegt ein neues Konzept zugrunde, das in [Levi2001] eingeführt wurde. Die Sensordaten werden im Fusionsagenten gemeinsamen Fusionsobjekten zugewiesen, wobei ein Fusionsobjekt als Resultat eines Fusionsvorgangs das Modell für ein reales Objekt in der Umwelt ist. Dabei lässt sich für jedes Objekt in der Umwelt ein separater Fusionsvorgang konfigurieren.

Für die Weltmodellierung wurden Klassen entworfen, welche die Straße, die Umweltbedingungen, die in der Umwelt befindlichen Objekte und die eigenen Fahrzeugdaten speichern und zur Weiterverarbeitung zur Verfügung stellen. Die Klassen wurden so konzipiert, dass Erweiterungen und Anpassungen in Zukunft leicht durchführbar sind. Neben der Datenhaltung verfügt der Weltmodellagent über Logik, die insbesondere die Verwaltung der Daten für die erkannten Objekte in der Umwelt übernimmt. Diese basiert auf einem in [Levi2001] vorgestellten Konzept zur Weltmodellierung, dem ein probabilistisches Netz mit topologischen Informationen zugrunde liegt. Das Weltmodell ist in der Lage, Objekte, die kurzzeitig von der Sensorik nicht erkannt werden, intern weiterzuführen. Darüberhinaus kann der anzunehmende Zustand eines Objekts zu einem späteren Zeitpunkt prädiziert werden. Zur Verbesserung der Weiterführung

und Prädiktion wurde eine Datenstruktur und Algorithmik entwickelt, die das Verhalten von Fahrzeugen beschreibt. Die gegenseitige Beeinflussung der beobachteten Verkehrsteilnehmer wurde dabei nicht berücksichtigt.

Es wurde im Rahmen dieser Arbeit eine Simulation angepasst, die als Sensordatenquelle für das entwickelte System diente. Die Schnittstellen zu den Sensoren wurden so flexibel gehalten, dass die Sensordatenquelle beispielsweise durch einen realen Versuchsträger ausgetauscht werden kann.

Das entwickelte System zeigt die grundsätzliche Machbarkeit und Brauchbarkeit der genannten Konzepte anhand einfacher Szenarien. Das Konzept des Multiagentensystems ist für Fahrerassistenzsysteme sehr nützlich, da auf geänderte Zielsetzungen und Bedingungen während der Laufzeit reagiert werden muss. Durch eine flexible Fusionsarchitektur für die Verarbeitung der Sensordaten kann man die Algorithmen dynamisch so strukturieren, wie sie in der aktuellen Verkehrssituation benötigt werden. Strukturiert man dabei für jedes erkannte Objekt in der Umwelt einen separaten Fusionsvorgang, sind optimale Konfigurationen und ein hoher Grad an Parallelität erreichbar.

Die Repräsentation des Wissens über die Eigen- und Umweltdaten in einem konsistenten und zuverlässigen Weltmodell ist eine wichtige Handlungsbasis für die darauf aufbauende Aktorik eines Fahrerassistenzsystems. Vermutungen über das Verhalten anderer Verkehrsteilnehmer zu treffen stellte sich als sehr nützlich für die realitätsnahe Weiterführung und Prädiktion heraus.

Fahrerassistenzsysteme werden in Zukunft anspruchsvolle Aufgaben in komplexen Verkehrsszenen zu bewältigen haben (z. B. automatischer Spurwechsel auf einer Autobahn in dichtem Verkehr). Dafür ist Multisensorik notwendig, um zuverlässige Informationen über die Umwelt des Fahrzeugs akquirieren zu können. Nach den Erkenntnissen dieser Arbeit kann die dadurch entstehende Komplexität nur durch flexible Systeme, in denen die Logik auf mehrere eigenständige, anpassbare Komponenten verteilt ist, beherrscht werden.

6.2 Ausblick

Abschließend werden nun einige Punkte genannt, die der weiteren Bearbeitung der in dieser Arbeit behandelten Thematik gelten. Dabei wird zuerst auf den Fusionsagenten, danach auf den Weltmodellagenten eingegangen. Am Ende sind noch Gedanken zum Gesamtsystem aufgeführt.

6.2.1 Fusionsagent

Im Rahmen dieser Arbeit wurde anhand einiger prototypischer Fusionsvorgänge die Verwendung einer flexiblen Fusionsarchitektur demonstriert und getestet. Der nächste Schritt zur Evaluation dieser Architektur muss die Einbettung realer Fusionsalgorithmen (Spurerkennung, Hinderniserkennung, etc.) sein. Zu berücksichtigen sind dabei die Fusionsmerkmale, die in dieser Arbeit vorgestellt wurden (Fusionsebenen, Fusionsarten, etc.). Diese ermöglichen eine nachvollziehbare Strukturierung der Sensordatenverarbeitung.

Weiterhin ist eine Algorithmik zu entwickeln, die selbständig die Konfiguration der Fusionsvorgänge übernimmt, je nach Aufgaben und Kontext des Systems. Erst dadurch kann die Autono-

mie des Fusionsagenten entstehen. Dabei ist es auch anzuraten, ein Modell des Sensornetzwerks im Fusionsagenten zu halten, das die aktuellen Zustände der Sensoren kennt und die Ansteuerung der Sensoren (z. B. An- und Ausschalten) vornimmt. Die Kommunikation zu den Sensoren könnte über die Sensorschnittstellen, die im Abschnitt 4.2.3 vorgestellt wurden, erfolgen.

6.2.2 Weltmodellagent

Zur Repräsentation und Berechnung der vom Versuchsträger beobachteten Fahrzeuge wurde in dieser Arbeit eine Klassenstruktur für ein probabilistisches Netz mit topologischen Informationen entworfen. In Zukunft muss untersucht werden, wie mit dieser Methodik nicht nur die Beziehung "anderes Fahrzeug - Versuchsträger", sondern auch die Beziehungen der anderen Fahrzeuge untereinander berücksichtigt werden können.

Die Berechnung der Wahrscheinlichkeiten für die im probabilistischen Netz gehaltenen Zustände beruht auf den Zuverlässigkeitswerten, die von den Sensoren für die akquirierten Merkmale vergeben werden, und auf Übergangswahrscheinlichkeiten von einem Zustand in einen anderen. Die Zuverlässigkeitswerte müssen über den Fusionsagenten in das Weltmodell gelangen. Die Übergangswahrscheinlichkeiten könnten u. a. durch Lernverfahren ermittelt werden.

Die in dieser Arbeit verwendeten Berechnungen zur Ermittlung von Zustandswerten müssen durch geeignetere Methoden (z. B. Kalman-Filter) ersetzt werden. Weiterhin müssen weitere Transaktionen für die Beschreibung des Verhaltens anderer Fahrzeuge integriert werden (z. B. Abbremsen, Abbruch eines Überholvorgangs, etc.).

Wurden Objekte in der Umwelt des Fahrzeugs erkannt, ist es relativ einfach, darauf zu schließen, das in dem Bereich der Umwelt etwas existiert. Es ist jedoch schwierig, Auskunft über Unwissenheit zu geben. D. h. wenn das Weltmodell über ein Gebiet in der Umwelt (z. B. ein "toter Bereich") keine Aussage treffen kann, ob es von einem Objekt belegt ist oder nicht, dann muss es dies "formulieren" können. Darüberhinaus gilt es, die Unwissenheit durch Inferenzen zu beseitigen.

6.2.3 Gesamtsystem

Das System muss zu einem Fahrerassistenzsystem vervollständigt werden, indem weitere Agenten (Situationsagent, Aktoren, etc.) hinzugefügt werden.

Der Austausch der Sensordatenquelle "Simulation" durch einen realen Versuchsträger wird erst zeigen können, wie sich das System im realen Einsatz verhält. Hier gilt es vor allem die Probleme der Echtzeitfähigkeit zu lösen.

Die Kommunikation zwischen den Agenten könnte durch Aufträge erweitert werden. Somit könnte z. B. der Fusionsagent den Sensoren Anweisungen geben, sich einzuschalten oder den Sehbereich zu ändern.

Der Test eines solchen flexiblen Systems, das in einer hoch dynamischen Umgebung Anwendung findet, wird sich als weitaus schwieriger gestalten als der Test eines statischen Systems. Ein flexibles System sollte in der Lage sein, selbständig auf Änderungen in der Umgebung, eigenen Sensorik oder der Aufgabenstellung durch Anpassungen seiner Komponenten zu reagieren.

Anhang A

Symbolische Postionsangaben

Für die Beschreibung der Positionen von Objekten um ein Fahrzeug herum, die von der physikalischen, metergenaue Position abstrahiert, wurden in dieser Arbeit symbolische Positionen definiert. Diese sind in der Abbildung A.1 eingezeichnet. Jede der symbolischen Positionen

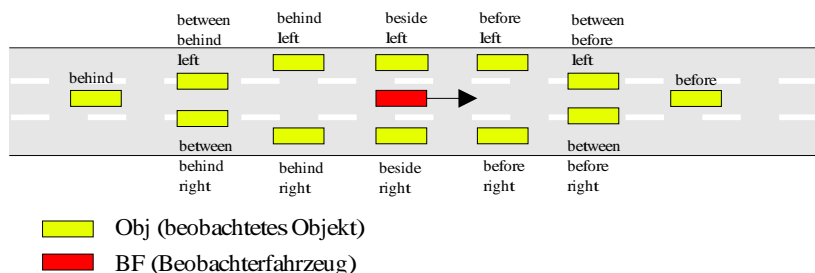


Abbildung A.1: Symbolische Positionen von Objekten

wird durch Bedingungen beschrieben, die erfüllt sein müssen, um sie einem Objekt zuordnen zu können. Diese Bedingungen werden im Folgenden angegeben, wobei **BF** für das Beobachterfahrzeug steht, relativ zu dem die symbolischen Positionen gelten. **Obj** ist das Symbol für das Objekt, dessen Position zu bestimmen ist.

BEHIND

- Obj ist auf der gleichen Spur wie BF
- keine Überlappung von Obj zur rechten oder linken angrenzenden Spur
- Vorderkante von Obj ist hinter der Hinterkante von BF

BEHIND-LEFT

- Obj ist eine Spur weiter links als BF

- keine Überlappung von Obj zur Spur von BF
- Vorderkante von Obj ist hinter der Hinterkante von BF

BEHIND-RIGHT

- Obj ist eine Spur weiter rechts als BF
- keine Überlappung von Obj zur Spur von BF
- Vorderkante von Obj ist hinter der Hinterkante von BF

BETWEEN-BEHIND-LEFT

- Obj ist eine Spur weiter links als BF oder auf der gleichen Spur wie BF
- Überlappung von Obj zur Spur von BF und zur linken Spur
- Vorderkante von Obj ist hinter der Hinterkante von BF

BETWEEN-BEHIND-RIGHT

- Obj ist eine Spur weiter rechts als BF oder auf der gleichen Spur wie BF
- Überlappung von Obj zur Spur von BF und zur rechten Spur
- Vorderkante von Obj ist hinter der Hinterkante von BF

BESIDE-LEFT

- Obj befindet sich links von BF
- Hinterkante von Obj ist hinter der Vorderkante von BF
- Vorderkante von Obj ist vor der Hinterkante von BF

BESIDE-RIGHT

- Obj befindet sich rechts von BF
- Hinterkante von Obj ist hinter der Vorderkante von BF
- Vorderkante von Obj ist vor der Hinterkante von BF

BEFORE, BEFORE-RIGHT, BEFORE-LEFT, BETWEEN-BEFORE-LEFT, BETWEEN-BEFORE-RIGHT

- Bedingungen der seitlichen Positionen analog zu den Bedingungen der Positionsangaben für den hinteren Bereich
- Hinterkante von Obj ist vor der Vorderkante von BF

Die aufgeführten Positionen müssen in Zukunft noch verfeinert und erweitert werden. Durch die Aneinanderreihung von räumlich nebeneinanderliegenden Positionen lassen sich Bewegungsabläufe von Objekten beschreiben.

Anhang B

Koordinatensysteme, relative Abstände und Winkel

Für jedes Objekt wurde ein Koordinatensystem definiert. Dadurch lässt sich der Winkel des Fahrzeugs relativ zum Koordinatensystem der Spur beschreiben. Weiterhin können die Abstände zwischen den Fahrzeugen definiert werden. Folgende Skizze gibt das Koordinatensystem und die Winkel eines Objekts relativ zur Spur aus der Vogelperspektive an. Der Nullpunkt des Koordina-

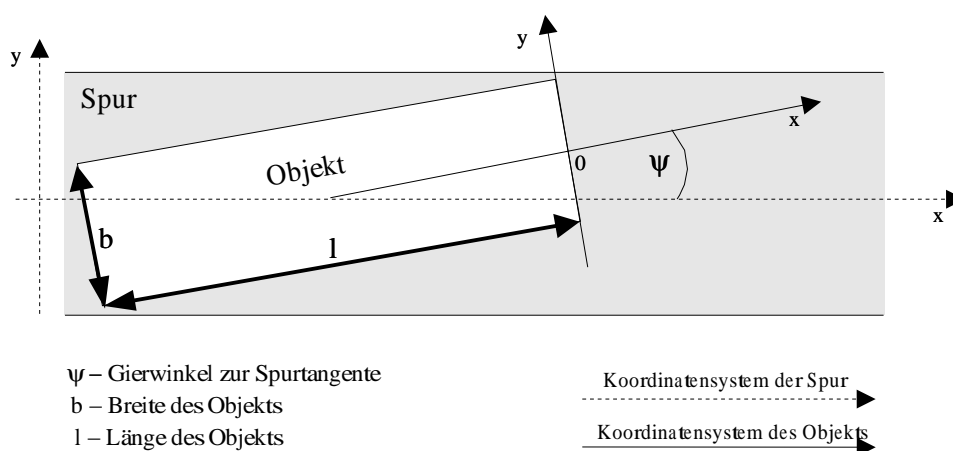


Abbildung B.1: Objekt aus der Vogelperspektive

tensystems eines Objekts wird an dessen Vorderkante positioniert. Die Seitenansicht eines Objekts stellt die Abbildung B.2 dar.

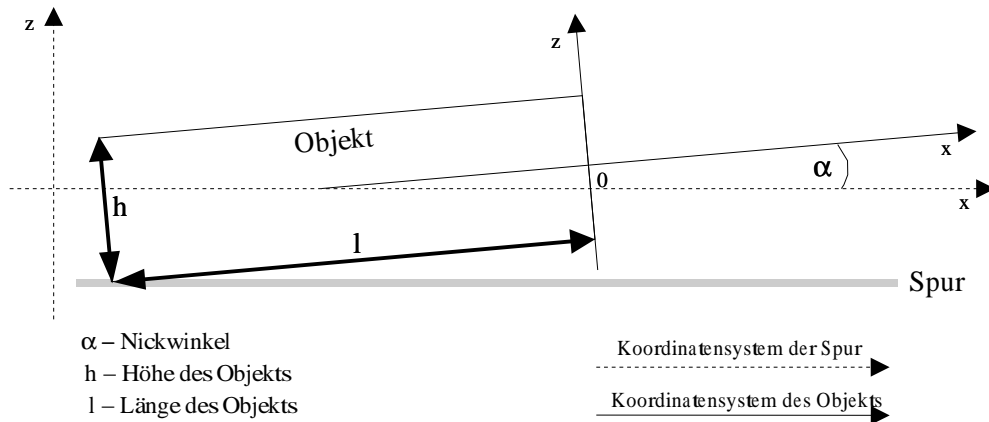


Abbildung B.2: Seitenansicht eines Objekts

Die Ansicht eines Objekts von hinten ist in der Abbildung B.3 skizziert.

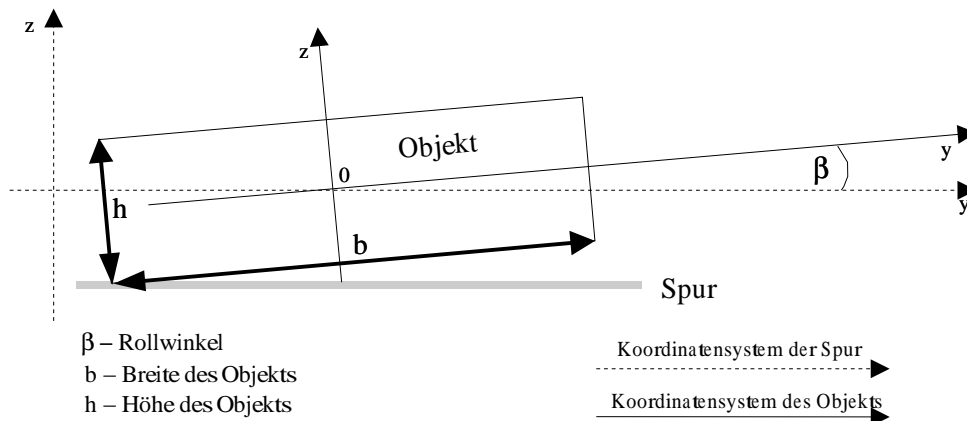


Abbildung B.3: Ansicht eines Objekts von hinten

Die Abstände zwischen Fahrzeugen wurden definiert durch den Abstand der Nullpunkte ihrer Koordinatensysteme bzgl. des Koordinatensystems der Spur. Die Abbildung B.4 skizziert den lateralen und longitudinalen Abstand zwischen zwei Objekten.

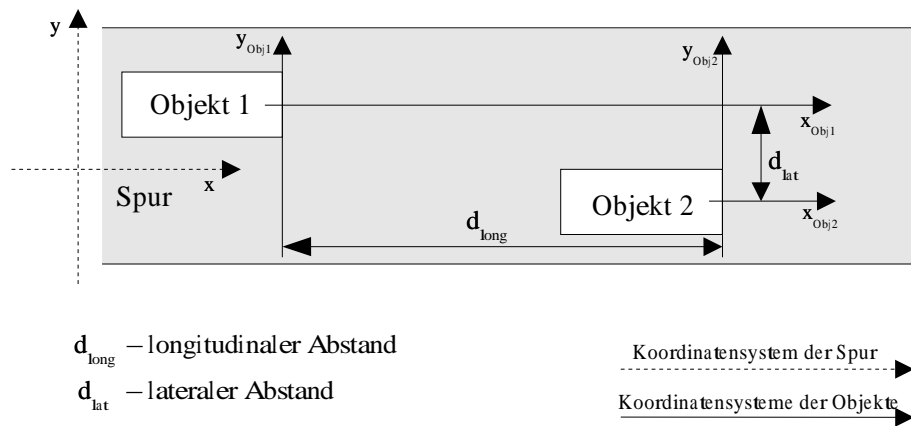


Abbildung B.4: Abstand zweier Objekte

Literaturverzeichnis

- [Brost2000] Brost, M.: *Einführung in Markov-Ketten und probabilistische Netze*. Studienarbeit. Universität Stuttgart. Dezember 2000
- [Gavrila99] Gavrila, D.: *Traffic sign recognition revisited*. Mustererkennung (Förstner, W., et al., eds.). Springer Verlag. 1999
- [Gern2000] Gern, A.; Franke, U.; Levi, P.: *Advanced Lane Recognition - Fusing Vision and Radar*. Proceedings of IEEE Conference on Intelligent Vehicles. 2000
- [Görzig98] Görzig, S.; Franke, U.: *ANTS - Intelligent Vision In Urban Traffic*. Proceedings of the 1998 IEEE International Conference on Intelligent Vehicles. S. 545-549. Oktober 1998
- [Janssen93] Janssen, R.; Ritter, W.; Stein, F.; Ott, S.: *Hybrid Approach for Traffic Sign Recognition*. Proceedings of Intelligent Vehicles Conference. 1993
- [Jörg91] Jörg, K.-W.: *Echtzeitfähige Multisensor-Integration für autonome mobile Roboter*. Reihe Informatik, Band 91. B.I. Wissenschaftsverlag. 1991
- [Levi2000] Levi, P.; Hetzel, G.; Lafrenz, R.; Schulé, M.: *Definition einer Software-Architektur für Weltrepräsentationen und Handlungsableitungen*. März 2000
- [Levi2001] Levi, P.; Breckle, H.; Bantle, A.: *Konzeption und Spezifikation von Fusionsobjekten als Schnittstelle zwischen der Fusionsebene und der Situationsinterpretation*. April 2001
- [May2000] May, F.: *Fahrassistenz*. Vortrag im Rahmen des 8. Seminar Elektronik im Kraftfahrzeugwesen. Esslingen. Januar 2000
- [Pearl88] Pearl, J.: *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann. San Mateo, Calif. 1988
- [Probst2001] Probst, J.: *Fusion von Radar und Bildverarbeitung für ein robustes Lokalisieren und Tracking von Fahrzeugen*. Diplomarbeit. Fachhochschule Stuttgart, Fachbereich Mathematik. März 2001

- [Reichardt96] Reichardt, D.: *Kontinuierliche Verhaltenssteuerung eines autonomen Fahrzeugs in dynamischer Umgebung*. Dissertation. Universität Kaiserslautern. 1996
- [Reimer91] Reimer, U.: *Einführung in die Wissensrepräsentation*. Leitfäden der angewandten Informatik. B. B. Teubner. Stuttgart. 1991
- [Schank77] Schank, R.C.; Abelson, R.P.: *Scripts, Plans, Goals and Understanding: An Inquiry into Human Knowledge Structures*. Lawrence Erlbaum. Hillsdale/N.J. 1977
- [Schiek2001] Schiek, S.: *Echtzeitfähige Repräsentation der Verkehrssituation durch Sensorfusion*. Diplomarbeit. Universität Mannheim, Technische Informatik. Juni 2001
- [Vlacic2001] Vlacic, L.; Parent, T.; Harashima, F.: *Intelligent vehicle technologies*. Butterworth Heinemann Verlag. 2001
- [Wachter2000] Wachter, M.: *Multisensorfusion zur Fahrzeug-Querführung unter Verwendung eines Fahrszenensimulators*. Diplomarbeit. Fachhochschule Stuttgart, Fachbereich Mathematik. März 2000
- [Zomotor97] Zomotor, Z.; Franke, U.: *Sensor fusion for improved vision based lane recognition and object tracking with range-finders*. Proceedings of IEEE Conference on Intelligent Vehicles. 1997

Tabellenverzeichnis

3.1	Merkmale einer Straße	24
3.2	Merkmale einer Spur	25
3.3	Eigene Fahrzeugdaten	26
3.4	Merkmale eines Objekts in der Umwelt	27
3.5	Merkmale des Kontextes	28
3.6	Merkmale der Straßenverkehrsordnung	28

Abbildungsverzeichnis

1.1	Bisherige Architektur eines Fahrerassistenzsystems	2
1.2	Grobe Architektur des zu entwickelnden Systems	3
2.1	Spurwechsel auf einer Autobahn	6
2.2	Komponenten von ANTS (FE = Funktionseinheit)	10
3.1	Fusionsebenen in einem Fusionsvorgang	15
3.2	Ausschnitt aus der Fusionsmerkmalsbibliothek	16
3.3	Darstellung einer Markov-Kette	18
3.4	Probabilistisches Netz	19
3.5	Grundform eines Dynamischen Bayes-Netz nach [Levi2001]	20
3.6	Einflüsse auf den Zustandsknoten S_t eines Fahrzeugs	21
3.7	Beispiel einer topologischen Karte, die auf einem DBN basiert	21
3.8	Versuchsträger mit 9 externen Sensoren	22
3.9	Klassendiagramm der Kontrollobjekte	31
3.10	Klassendiagramm der Fusionsobjekte	33
3.11	Beispielhafter Fusionsvorgang	34
4.1	Einbettung des zu entwickelnden Systems	38
4.2	Architektur des Systems	39
4.3	Funktionseinheiten zur Beschaffung der Sensordaten	40
4.4	Aktionsliste für einen Überholvorgang	42
4.5	Hinzugefügte Sensoren	43
4.6	Grobe Struktur einer Sensorschnittstelle	44
4.7	Grobe Struktur des Fusionsagenten	46
4.8	Fusionsvorgang für ein Objekt in der Umwelt	48
4.9	Aufgaben und Systemeinbettung des Weltmodellagenten	52
4.10	Klassendiagramm des Weltmodells	54

4.11	Attribute eines Merkmals	55
4.12	Klassendiagramm der eigenen Daten	56
4.13	Klassendiagramm des Kontextes	59
4.14	Attribute eines Objekts in der Umwelt	61
4.15	Klassendiagramm des Topologischen Netzes	62
4.16	Klassendiagramm der Bibliothek	64
4.17	Transaktion für das Überholmanöver 1	64
4.18	Transaktion für das Überholmanöver 2	65
4.19	Transaktion für das Überholmanöver 3	65
4.20	Bereiche, die nicht von der Sensorik abgedeckt werden	66
4.21	Klassendiagramm des Bayes-Netzes	66
4.22	Beispiel eines Bayes Netzes mit topologischen Informationen	69
5.1	Szenario 1	73
5.2	Szenario 2	73
5.3	Szenario 3	73
5.4	Drei dynamische Bayes-Netze mit unterschiedlichen Zeitabständen	78
5.5	Prädiktion unter Berücksichtigung zwei möglicher Transaktionen	79
5.6	Vergleich von Schätzung und Beobachtung im Sehbereich der linken Spiegelkamera (in der Skizze a ist die linke Spiegelkamera angeschaltet, in der Skizze b müssen die Werte innerhalb des fehlenden Sehbereichs geschätzt werden)	80
A.1	Symbolische Positionen von Objekten	86
B.1	Objekt aus der Vogelperspektive	88
B.2	Seitenansicht eines Objekts	89
B.3	Ansicht eines Objekts von hinten	89
B.4	Abstand zweier Objekte	90

Erklärung

Ich versichere, dass ich diese Arbeit selbständig verfasst und nur die angegebenen Hilfsmittel verwendet habe.

André Schammer

Esslingen, den 6. September 2001