Humboldt-Universität zu Berlin
Mathematisch-Naturwissenschaftliche Fakultät
Institut für Informatik

Multi-Hypothesen-Partikelfilter zur Objektmodellierung am Beispiel eines Tormodells im RoboCup-Kontext

Diplomarbeit

zur Erlangung des akademischen Grades
Diplominformatiker

eingereicht von: Marcus Scheunemann
geboren am: 15.01.1985
geboren in: Berlin

Gutachter/innen: Prof. Dr. Hans-Dieter Burkhard
Prof. Dr. Verena Hafner

eingereicht am: ..... verteidigt am: .....

Information about objects and their positions in an environment are necessary requirements for most tasks of a mobile autonomous robot, in particular regarding the control of behaviour and navigation. This presents a special challenge for robots with a limited view angle.

Autonomously soccer playing robots in the RoboCup Standard Platform League are exposed to these difficulties in a dynamic environment. Most approaches aggregate all available information in one holistic model in order to localise robots. In case of inconsistent perceptions the model either turns noisy or opens and tracks a further hypothesis. To improve the localisation, and thus the behaviour control, local models have received only little attention so far.

In this work the implementation of a local goal model is presented and analysed. A multi-hypothesis particle filter is used to deal with the ambiguity of goal post percepts as well as to process incomplete and uncertain sensor information. Additionally, a percept puffer supports the initialisation and also facilitates the handling of sparse false measurements. On the basis of this local goal model inconsistencies can be explicitly modeled, which may be used to stabilise the location of a robot because explicitly modeled inconsistencies are not aggregated.

To accomplish this task, but especially with regard to future modeling tasks, this work additionally creates an overview of error sources in object recognition and odometry calculations of the underlying NaoTH framework.

Die meisten Aufgaben eines mobilen autonomen Roboters, insbesondere die der Verhaltenssteuerung und Navigation, setzen Informationen über Objekte und deren Position in der Umgebung voraus. Bei Robotern mit einem beschränktem Sichtfeld stellt dies eine besondere Herausforderung dar.

Autonom fußballspielende Roboter der RoboCup Standard Platform League sind diesen Herausforderungen in einer dynamischen Umgebung ausgesetzt. Die meisten Ansätze zur Lokalisierung der Roboter aggregieren alle zur Verfügung stehenden Informationen in einem ganzheitlichen Modell. Bei inkonsistenten Wahrnehmungen verrauscht entweder dieses Modell, oder das Modell eröffnet und pflegt eine weitere Hypothese. Zur Verbesserung der Lokalisierung, und damit auch der Verhaltenssteuerung, finden lokale Modelle bisher jedoch nur geringe Beachtung.

In dieser Arbeit wird die Implementierung eines lokalen Tormodells vorgestellt und analysiert. Ein Multi-Hypothesen-Partikelfilter wird verwendet, um mit der Mehrdeutigkeit von Torpfostenperzepten umzugehen sowie unvollständige und unsichere Sensorinformationen zu verarbeiten. Zusätzlich dazu unterstützt ein Perzeptpuffer die Initialisierung und erleichtert darüber hinaus den Umgang mit spärlichen Falschmessungen. Anhand dieses lokalen Tormodells können Inkonsistenzen explizit modelliert werden, was wiederum dafür verwendet werden kann, die Lokalisierung eines Roboters zu stabilisieren, indem man explizit modellierte Inkonsistenzen nicht aggregiert.

Zur Erfüllung dieser Aufgabe, vor allem aber auch in Hinblick auf zukünftige Modellierungsaufgaben, erstellt diese Arbeit zusätzlich eine Übersicht von Fehlerquellen der Objekterkennung und Odometrieberechnungen des zugrunde gelegten NaoTH-Frameworks.

Inhaltsverzeichnis

1 Einleitung

Die meisten Aufgaben eines mobilen Roboters, insbesondere die der Verhaltenssteuerung und Navigation, setzen Informationen über Objekte und deren Position in der Umgebung voraus. Dies stellt vor allem für Roboter, die mit einer Kamera mit beschränktem Öffnungswinkel ausgestattet sind, eine zusätzliche Herausforderung dar.

Die lückenhaften und verrauschten Sensorinformationen führen zu einer unsicheren Vorstellung der Umgebung. Ein geeignetes Modell der umliegenden Objekte ist notwendig, um den Roboter das Erarbeiten von Plänen zu ermöglichen und so komplexes Verhalten zu verwirklichen.

Der RoboCup bietet zur Untersuchung dieser Fragestellungen ein sehr gutes Testumfeld. In der Standard Platform League des RoboCups spielen jeweils zwei autonome Robotermannschaften gegeneinander Fußball. Gerade diese Liga bietet ein optimales Umfeld zum Untersuchen und Vergleichen von Algorithmen, da alle Mannschaften aus nahezu baugleichen Robotern zusammengestellt sind.[1] Die Struktur der Umgebung ist dabei fest beschrieben, so dass speziell die Positionen von Linien und Toren bekannt sind. Daher ist der Ansatz der meisten Teams, dass die Roboter ihr Verhalten anhand ihrer ermittelten globalen Position steuern. Zur Berechnung dieser Position verwenden die meisten Arbeitsgruppen ein Modell, welches Wissen über die festen geometrischen Feldgrenzen einbezieht. Hierzu werden viele der zur Verfügung stehenden Informationen (z.B. Linien, Tor und Umfeld) aggregiert. Allerdings können sich verändernde Lichtverhältnisse oder Objekte außerhalb des Spielfeldes zu Inkonsistenzen der Roboterwahrnehmung führen, so dass der modellierte Zustand dadurch verrauscht werden kann.

Ein lokales Tormodell kann solche Inkonsistenzen explizit modellieren, gesehene einzelne Torpfosten klassifizieren und so den Rechenaufwand zur Konsistenzerhaltung eines globalen Modells verringern. Darüber hinaus kann ein lokales Tormodell einem Roboter bereits den erfolgreichen Torabschluss ermöglichen, auch wenn dieser keine Kenntnisse mehr über seine genaue Spielfeldposition hat. Wenn ein Roboter beispielsweise den Ball sieht und ein Modell über das gegnerische Tor gepflegt hat, kann er schon allein aufgrund dieser Relation einen groben Verhaltensplan ausarbeiten.

[1] Beim RoboCup ist es eine der Hauptaufgaben vieler Mannschaften, innerhalb unterschiedlicher Ligen ein Fußballspiel zu bestreiten. Eine detaillierte Beschreibung folgt im Abschnitt 2.1.

Die Sensoreingaben, die zur Bildung dieses Modells führen, müssen aufgrund des Umweltrauschens und der Roboterbewegung gefiltert werden. So wird bereits durch die Bewegung des Roboters eine verrauschte Sensoreingabe begünstigt. Zusätzlich dazu muss das bisherige Wissen von Objekten, die sich aktuell nicht im Sichtfeld befinden, in irgendeiner Form korrespondierend zur Eigenbewegung *mitverschoben* werden. Eine besondere Schwierigkeit dieser Modellierungsaufgabe ergibt sich durch andere Roboter auf dem Feld, da diese Teile eines relevanten Objekts verdecken können. Es kann also nicht davon ausgegangen werden, dass bei einem Blick in die Richtung des Modells auch tatsächlich das modellierte Objekt zu sehen ist. Weiterhin ist zu berücksichtigen, dass mehrere Objekte gleicher Form, wie zum Beispiel Torpfosten und Roboter, auf dem Spielfeld vorhanden sind, was wiederum zu einem Assoziationsproblem führt.

Zur besseren Einordnung dieser Arbeit werden im folgenden Kapitel neben der nachfolgenden Motivation auch der Anteil des Autors aufgezeigt sowie die Zielstellung konkretisiert.

1.1 Motivation

Genau wie wir Menschen können auch die Roboter während des Spiels nicht das gesamte Feld überblicken, denn ihre Wahrnehmung ist beschränkt. Besonders relevant ist diese Erkenntnis, wenn ein Roboter beispielsweise direkt vor einem Tor steht und das Erblicken beider Pfosten in einem Kamerabild nicht mehr möglich ist. Die Abbildung 1.1 zeigt ein Beispiel einer solchen Spielsituation. Das linke Bild zeigt die Roboterperspektive. Der Roboter sieht einen Ball, einen Pfosten und eine Linie. Legt man nun lediglich dieses Bild zugrunde, dann ist für den Roboter nicht ersichtlich, ob es sich um den rechten oder linken Pfosten handelt.

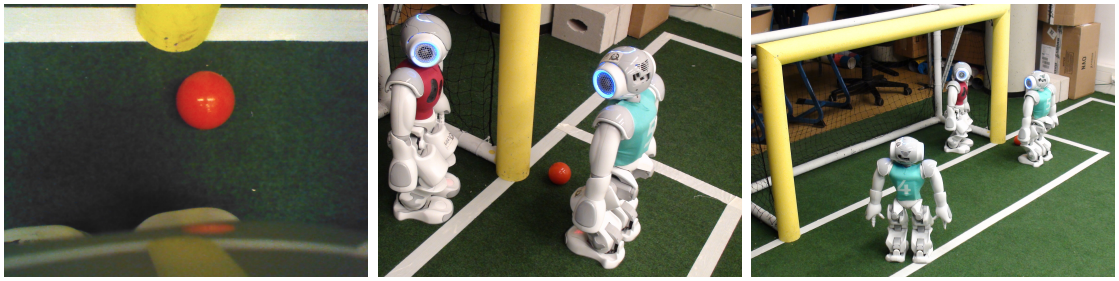

Abbildung 1.1: *Links: Die Sicht des angreifenden Roboters. Er kann den Ball und den Teil eines Pfostens erkennen. Mitte: Die tatsächliche Situation des Roboters. Es wird klar, dass der Roboter auf den rechten Torpfosten schaut. Rechts: Ein weiterer Mitspieler steht zur Linken des angreifenden Roboters. Ein Passspiel ist womöglich die beste Option.*

Das mittlere Bild zeigt indes einen größeren Ausschnitt der Spielsituation aus einer *globaleren* Perspektive, aus welcher hervorgeht, dass der Roboter vor dem rechten Torpfosten steht. Zur Maximierung der Torchance muss der Roboter also nach links schießen. Eine weitere Perspektive ist im rechten Bild dargestellt. Durch ein erfolgreiches Passspiel zum Mitspieler wird die Torchance nochmals erhöht.

Diese Sequenzen zeigen, dass ein Roboter *Wissen* aus vergangenen Situationen integrieren sollte, um möglichst effizient zu entscheiden. Wenn der betrachtete Roboter in der Vergangenheit nach links geschaut hat, so hat er womöglich den linken Torpfosten wahrgenommen. Dieses Wissen könnte zur Lösung der aktuellen Situation hilfreich sein. Es ist dabei aber unklar, wie *sicher* sich der Roboter über die Position des linken Pfostens sein kann. Auch wenn ein Pfosten ein statisches Objekt ist, so ist es der Roboter selbst nicht. Er kann sich nach seiner letzten Pfostensichtung bewegt haben oder aber seine Wahrnehmung des linken Pfostens war schlichtweg ungenau.

Modelle spielen eine wesentliche Rolle bei der Lösung dieses Problems. Sie bieten die Grundlage für komplexe spielentscheidende Pläne. Bei wachsender Komplexität der Aufgaben muss hierbei immer ein Augenmerk auf die Performanz gelegt werden. Aufgrund der Tatsache, dass die Roboter autonom agieren, muss die Komplexität der gewünschten Modelle den vorhandenen limitierten Ressourcen entsprechen.

Effiziente Modellierungen zur Beherrschung komplexer Spielsituationen sind wichtig, um im Rahmen des RoboCups ein effizientes Roboterverhalten zu erzeugen. Dabei kann es ein entscheidender Vorteil sein, wenn sich Roboter besonders der Verarbeitung aktuell relevanter Informationen widmen und andere Berechnung nur *grob* weiterführen. Sollte der Roboter die Lage des Tores *kennen* und er sich gerade als ballführender Spieler im Angriff befinden, so ist unter Umständen die exakte Position auf dem Feld irrelevant und die Ermittlung jener kostet lediglich Zeit und Rechenkapazität.

Um ein solches Verhalten zu erforschen ist es zwingend notwendig, dass gut ausgearbeitete Modelle einzelner Objekte zur Verfügung stehen. Die Domäne des RoboCups bietet sich für die Untersuchung solcher Modelle an.

1.2 Zielstellung

Das zentrale Ziel dieser Arbeit sind Entwicklung und Implementierung eines geeigneten Modells für das Fußballtor zur Verarbeitung von verrauschten mehrdeutigen Torpfostenperzepten. Basierend auf der Idee der Bayes'schen Modellierung soll ein Multi-Hypothesen-Modell umgesetzt werden, und seine grundlegende Funktionsweise empirisch aufgezeigt werden. Dabei soll sowohl auf die schon vorhandene Verarbeitung visueller Informationen als auch auf bereits berechnete Odometriedaten

aufgebaut werden. Beide Informationsquellen gelten dabei als *Blackbox*. Gleichwohl soll diese Arbeit einen Überblick über das zugrunde liegende Sensorrauschen des Roboters vermitteln. Dazu sollen diverse Fehlerquellen aus der Odometrieberechnung sowie der visuellen Wahrnehmung exemplarisch aufgezeigt werden.

Als sekundäres Ziel hat dieses Werk den Anspruch, eine Grundlage für weitere Arbeiten speziell zum Thema der Bayes'schen Modellierung im RoboCup zu dienen. Dieser Anspruch spiegelt sich sowohl in der systematischen Struktur als auch in dem betont didaktisch aufgearbeiteten Grundlagenteil zur Bayes'schen Modellierung wieder.

1.3 Vorgehensweise

Zur Erarbeitung der genannten Ziele ist diese Arbeit folgendermaßen gegliedert. Das Arbeitsumfeld dieser Arbeit wird in Kapitel 2 vorgestellt. Das Kapitel 3 erarbeitet die Grundlagen zur Bayes'schen Modellierung von Objekten. Als eine mögliche Approximation dieser Modellierung wird der Partikelfilter diskutiert.

In Kapitel 4 sind relevante Forschungsergebnisse aus verschiedenen Disziplinen zusammengetragen. Ebenso zieht der Autor in jenem Kapitel eine Schlussfolgerung zur Wahl der grundlegenden Algorithmen.

Der aus den bis hierhin präsentierten Analysen und Vorwissen resultierende Implementierungsvorschlag wird in Kapitel 5 detaillierter beleuchtet.

Die Ungenauigkeiten der Objekterkennung werden in Kapitel 6 empirisch aufgezeigt. Kapitel 7 widmet sich der Analyse bezüglich des Erkennens der Eigenbewegung. Auch hier werden mögliche Ungenauigkeiten empirisch veranschaulicht.

Die Funktionsweise der Implementierung des vorgestellten Algorithmus wird daraufhin in Kapitel 8 experimentell untersucht.

In Kapitel 9 folgen eine Zusammenfassung der aus dieser Arbeit resultierenden Erkenntnisse sowie ein Ausblick für notwendige Folgeuntersuchungen. Darüber hinaus werden Verwendungsfelder des Multi-Hypothesen-Tormodells beleuchtet.

1.4 Beitrag des Autors

Der Autor ist seit 2010 Mitglied der Arbeitsgruppe Nao Team Humboldt. Seit 2011 ist er darüber hinaus gleichberechtigter Teamleiter des Nao Team Humboldts zusammen mit Heinrich Mellmann. Neben zahlreichen organisatorischen Tätigkeiten bestand ein großer Teil seiner Arbeit darin, die ganzheitliche Strategie der Arbeitsgruppe abzustimmen. Darüber hinaus unterstützte der Autor andere Studenten bei ihren Arbeiten innerhalb der Arbeitsgruppe, so dass er sich an Tests und Analysen in allen Aufgabenbereichen beteiligte, welche die Framework-Entwicklung, den

Roboterlauf, die Verhaltenssteuerung und die Modellierung umfasst. Eigene Implementierungen des Autors sind besonders im Bereich der Verhaltenssteuerung und Modellierung auszumachen. Er implementierte unter anderem das Torwartverhalten, Teamstrategien bei Standardsituationen und das Verhalten eines Angreifers; in der Modellierung beschäftigte sich der Autor beispielsweise mit dem Ballmodell, einem kollaborativen Ballmodell aller Roboter, der Pfadplanung sowie Gegnererkennung. Des Weiteren arbeitete er an der Umsetzung eines Simulators mit den Regeln der RoboCup-Standardplattformliga zur besseren Analyse des Roboterverhaltens.

Bereits beim RoboCup 2010 wurden Teile seiner Arbeit verwendet. Persönlich nahm der Autor an den RoboCups 2011, 2012, 2013 und 2014 teil und konnte seine Beiträge dazu leisten, dass sich das Team jährlich verbesserte.

Zur Förderung des Wissensaustausches zwischen den einzelnen deutschen Arbeitsgruppen richtete er gemeinsam mit Heinrich Mellmann eine Workshopreihe für RoboCup-Teams (kurz: RoboW) ein. Er organisierte diesen Workshop mehrmalig in Berlin. Heutzutage findet dieser regelmäßig zwei- bis dreimal pro Jahr statt.

Die Arbeiten für diese Diplomarbeit begannen bereits 2011. Bis zur derzeitigen Fertigstellung des Tormodells haben diverse Analysen und Experimente stets Rückschlüsse und Verbesserung der Infrastruktur, insbesondere der Bildverarbeitung, nach sich gezogen. Das Multi-Hypothesen-Tormodell ist in C++ vom Autor implementiert worden. Darüber hinaus sind weitere Werkzeuge und Klassen entstanden, um anknüpfende Modellierungsaufgaben mit Partikelfiltern angehen zu können.

1.5 Veröffentlichungen

Zusätzlich zur Teamarbeit beschäftigte sich der Autor parallel zum RoboCup mit der Umsetzung eines adaptiven Greifalgorithmus für den Nao [MSS13] und unterstützte die Veröffentlichung zur strategischen Positionierung von Mitspielern im RoboCup [KMSB13].

Eine erste Implementierung des Algorithmus dieser Arbeit wurde 2011 vorgestellt [MS11].

Darüber hinaus schrieb er an den in 2011, 2012, 2013 und 2014 erschienenen *Team Desription Papers* mit [MSBH14].

2 Arbeitsumfeld und technische Voraussetzungen

Dieses Kapitel beleuchtet das Arbeitsumfeld, welches zur Erarbeitung des Themas zur Verfügung steht. In Abschnitt 2.1 wird der RoboCup erläutert. Bei solchen Wettstreits treten unter anderem mehrere Roboterfußballteams gegeneinander an. Aufgrund der Teilnahme am RoboCup konnte der Autor innerhalb der letzten Jahre an einem geeigneten Tormodell arbeiten, wobei sich besonders die vielen Test- und Turnierspiele für die Feldbeobachtung von Problemen eignen.

Der autonome humanoide Roboter Nao des Unternehmens Aldebaran Robotics, der innerhalb der Standard Platform League eingesetzt wird, dient dem Autor dabei als Untersuchungsgegenstand. Details zur Plattform werden in Abschnitt 2.2 besprochen.

Innerhalb des RoboCup verwendet des Autors Arbeitsgruppe, das Nao Team Humboldt, das NaoTH-Framework. Dies ist eine Software, die innerhalb der Teamarbeit entstand und vielfältige Möglichkeiten zur Arbeit mit den Robotern bietet. Diese drei wesentlichen Teile – der RoboCup, die Roboterplattform Nao sowie das Framework NaoTH – werden in den nachfolgenden Absätzen beschrieben. Der Abschnitt 2.3 umreißt die verwendeten Möglichkeiten dieser Software und gibt einen grundlegenden Überblick zur vorhandenen Objekterkennung.

Der letzte Abschnitt befasst sich mit dem Trackingsystem OptiTrack [OpT14]. Dieses System erfasst das Fußballfeld im Labor und dient dem Autor zur Überprüfung seiner Implementierung.

2.1 RoboCup

Das Langzeitziel des RoboCups ist es, im Jahr 2050 mit einer Roboterfußballmannschaft gegen den amtierenden Fußballweltmeister zu gewinnen. Auf dem Weg, dies zu verwirklichen, werden Erkenntnisse aus verschiedenen Forschungszweigen zusammengeführt. Die RoboCup-Initiative wurde im Jahre 1997 gegründet und trägt jährlich eine Weltmeisterschaft aus, an der Studenten und Wissenschaftler verschiedener Nationen teilnehmen. Zudem werden auch einige lokale Meisterschaften aus-

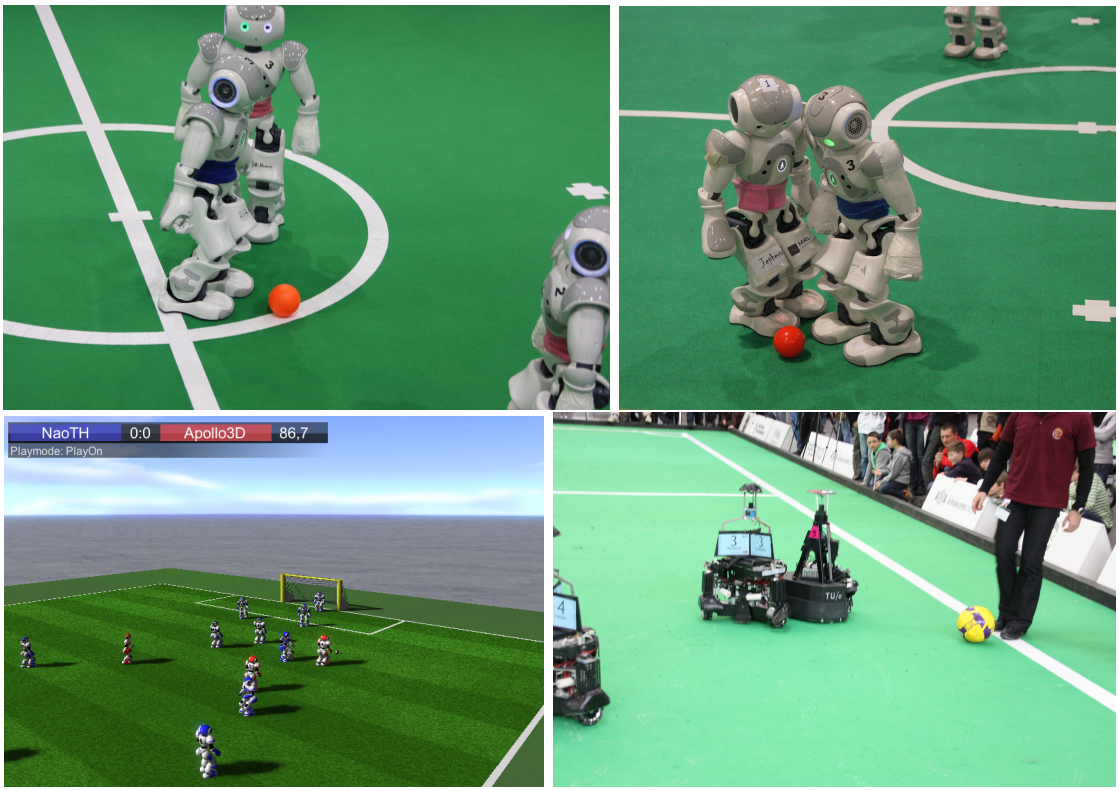

Abbildung 2.1: *Oben: Spielszenen innerhalb eines Spiels in der „Standard Platform League“ (SPL). Roboter unterschiedlicher Mannschaften interagieren mit dem Ball. Unten links: Eine Szene aus einem Spiel der „3D Simulation League“. Unten rechts: Die „Middle Size League“ mit fahrenden Robotern. Als Spielgerät dient hier ein herkömmlicher Fußball.*

getragen und Workshops[1] durchgeführt. Die größte dieser lokalen Meisterschaften ist die *GermanOpen*. Diese findet jährlich in Magdeburg statt. Die Organisatoren des *RoboCup*s beschreiben den Wettbewerb wie folgt:

> „RoboCup ist eine internationale Initiative mit dem Ziel, die Weiterentwicklung intelligenter Roboter durch die Veranstaltung von Wettbewerben zu fördern. Die Wettbewerbe dienen den Wissenschaftlern und Studenten als publikumswirksames Testfeld zur Demonstration der Fähigkeiten heutiger Roboter“ [RCP11] [RCD13].

In vielen unterschiedlichen Ligen mit verschiedenen Schwerpunkten versucht man diesen Ansprüchen gerecht zu werden. Eine Hauptdisziplin stellt dabei traditionell

[1] Der Autor ist Mitinitiator und mehrfacher Organisator des *Robotic Workshops*, (*RoboW*). Dieser Workshop findet zwei bis drei Mal jährlich statt.

der Fußball dar. Allein dieses Betätigungsfeld besteht aus einer Vielzahl unterschiedlichster Ligen mit verschiedenen Interessensbereichen.

So existieren reine Simulationsligen, deren Hauptaugenmerk auf das Erforschen von Verhalten gerichtet ist. Ebenso gibt es verschiedene Ligen, in denen Humanoide unterschiedlicher Größe ihre Fähigkeiten gegeneinander messen. Auch fahrende Roboterplattformen sind vertreten.

Die Abbildung 2.1 zeigt drei verschiedene Ligen des *RoboCup*s. Die oberen Graphiken zeigen die *Standard Platform League* (SPL). Das Bild unten Links der Abbildung 2.1 zeigt die *3D Simulation League* . Beide Ligen zeichnen sich dadurch aus, dass alle Teams die gleiche Roboterplattform nutzen.

In der Simulationsliga nutzt man eine virtuelle Wahrnehmung. Den einzelnen Robotern werden Informationen über die Position von Objekten in ihrem Sichtfeld bereitgestellt. Darüber hinaus sind die Agenten exakt baugleich und haben keinerlei Verschleißerscheinungen. Daher kann man sich in dieser Liga hauptsächlich der Strategieverbesserung und dem Schwarmverhalten widmen. Abbildung 2.1 (unten links) zeigt einen Ausschnitt aus der 3D-Simulationsliga.

In der Middle Size League (MSL) und Small Size League (SSL) kommen Roboter auf omnidirektionalen Rädern zum Einsatz. Einen Eindruck verschafft die Abbildung 2.1 (rechts unten). Die oftmals eingesetzten 360°-Kameras in der MSL führen zu weiteren Vorteilen. Durch das nahezu nahtlose Wissen bezüglich der Position des Balls und der stabilen Lage im Raum, können sich die Arbeitsgruppen auf strategische Fragen der Verhaltenssteuerung konzentrieren. Modellierungsfragen stellen sich nur eingeschränkt. So ist die Umsetzung der Lokalisierung durch den Einsatz solcher Kameras stark vereinfacht. Ein Rundumbild enthält in jedem Zeitschritt sehr viele Informationen. Diese sind in den meisten Fällen für die Berechnung der eigenen Position ausreichend. Dabei muss der Roboter kein vergangenes Wissen integrieren.

Die Roboter der SSL werden darüber hinaus von einem zentralen Computer gesteuert. Dieser kann auf die globale Position der Roboter zugreifen.

Das Nao Team Humboldt [Nao13], die Arbeitsgruppe in dessen Umfeld diese Arbeit entstand, nimmt an der Standard Platform League (SPL) [Tec11] teil. Innerhalb dieser Liga nutzen alle Teams Roboter gleicher Bauart, an denen keine technischen und nur minimale optische Veränderungen vorgenommen werden dürfen.

Zunächst bediente man sich dafür der Roboterplattform AIBO, ein hundähnlicher Roboter von Sony. Durch die Produktionseinstellung dieser Plattform entschied man sich 2008 zur Umstellung der Liga auf den humanoiden Roboters Nao von Aldebaran Robotics. Im Abschnitt 2.2 wird dieser Roboter genauer vorgestellt. Innerhalb der SPL liegt demnach der Fokus auf dem Vergleich von Algorithmen

und Methoden. Besonders relevant ist die Bildverarbeitung, das Erzeugen von Bewegungen und die Verhaltenssteuerung von Robotern im Schwarm.

Die Roboter spielen in zwei Mannschaften. Zum Zeitpunkt der Veröffentlichung dieser Arbeit (2014) besteht eine Mannschaft aus fünf Spielern. Wie in allen Ligen werden die Regeln vor jedem Turnier genau festgelegt[2]. Gespielt wird in dieser Liga auf einem Fußballfeld mit fest definierten Abmaßen. Das Feld aus Abbildung 2.2 zeigt die offiziellen Abmessungen bis zum Jahr 2012. Der Ball, die Tore, das Feld und die Linien unterliegen ebenso festen Abmaßen und genügen groben Farbvorgaben. Die Beleuchtung unterliegt hingegen keinen besonderen Vorgaben. Es gibt jedes Jahr neue Regeländerungen um die Liga schrittweise umgebungsunabhängig zu gestalten.

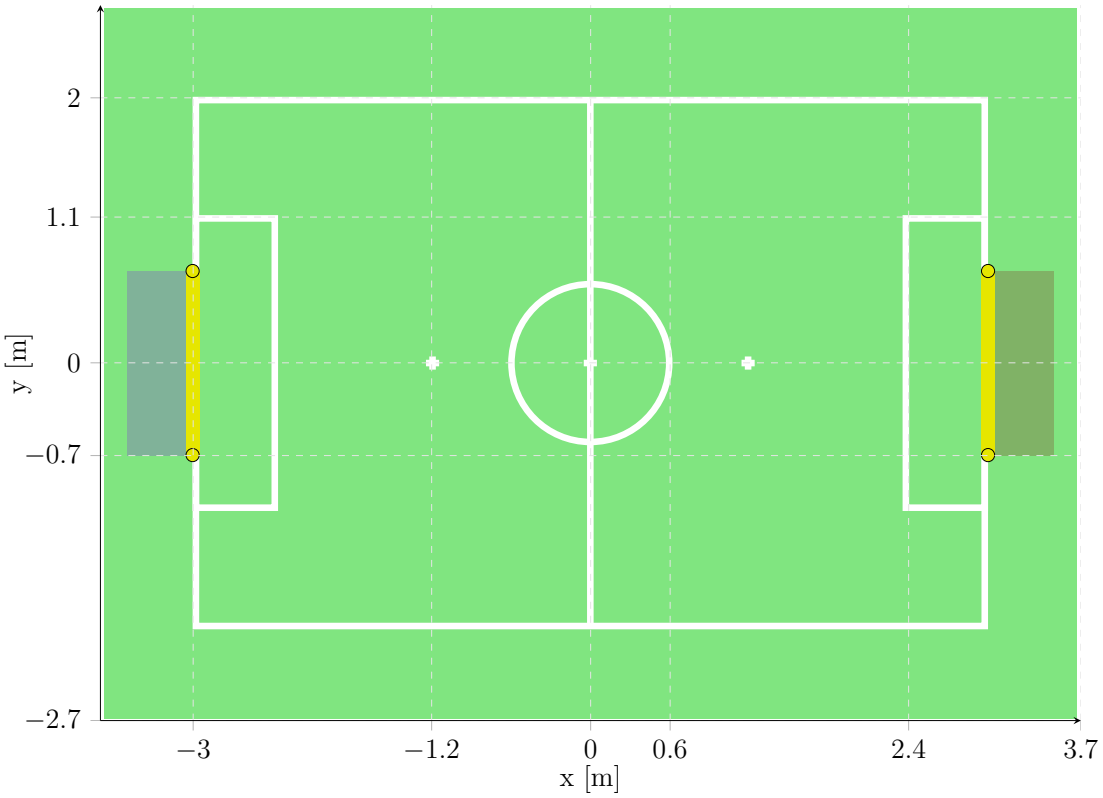

Abbildung 2.2: *RoboCup-Feld der* Standard Platform League *nach den Regeln von 2012. Seit 2013 hat sich das Feld bei gleichbleibender Strafraum- und Anstoßkreisgröße um das 1,5-fache vergrößert ($x = 9\,\mathrm{m}$, $y = 6\,\mathrm{m}$).*

Die oberen Bilder der Abbildung 2.1 zeigen Spielszenen während eines Spiels der SPL. Die ungeregelten Lichtverhältnisse resultieren in Schattenwürfen. Eine gute

[2]Unter [**?**] können die jeweils aktuellen Regeln der *Standard Platform Legaue* eingesehen werden.

Bildverarbeitung kann sich diesem Umstand anpassen. Im Kapitel 6 wird genauer darauf eingegangen, welche Probleme die Bildverarbeitung zu lösen hat. Es gibt viele weitere Arbeitsgruppen die ebenfalls einen hohen Beitrag zum Fortschreiten der Problemlösungsstrategien in der Liga liefern. Beispielhaft sei an dieser Stelle B-Human [RLM+13], die Nao Devils [CKU+10] und rUNSWift [AH14] und Austin Villa [BGH+13] genannt.

Obgleich derzeit in der SPL bei den meisten Mannschaften ein stabiler Gang zu beobachten ist, so ist diese Art der Fortbewegung grundsätzlich labiler als die eines fahrenden Roboters. Die Roboter können umfallen oder umgestoßen werden. Aber auch durch Berührungen oder rutschen auf dem Boden können die Roboter ihre Lage im Raum verändern, ohne dass dies durch Motorsteuerung in tatsächliche Positionsdaten zurückgerrechnet werden kann. Dies wird von Kapitel 7 detailliert untersucht.

Dem Abschnitt 2.2 ist ferner zu entnehmen, dass der Nao ein beschränktes Sichtfeld hat. Die grundsätzlich beschränkten Wahrnehmungsmöglichkeiten führen zur Notwendigkeit, in irgendeiner Form Wissen über die Zeit zu konservieren um angebrachte Verhaltensstrategien zu aktivieren. Ein Roboter muss daher aktiv seine Umgebung erkunden um wichtige Elemente, wie beispielsweise den Ball, *im Auge zu behalten*. Ohne das Wissen seiner Position und der Ballposition ist eine kooperative Arbeit im Schwarm nur schwer vorstellbar. Um die Möglichkeiten und Notwendigkeiten genauer abzuschätzen, beschreibt der nachfolgende Abschnitt die Roboterplattform Nao detailliert.

2.2 Roboter-Plattform Nao

Der humanoide Roboter Nao wird von der französischen Firma Aldebaran Robotics hergestellt [Ald13]. Es gibt mehrere Versionen und Ausführungen. Die Bauform H25 hat eine Größe von 58 cm und wiegt inklusive der Batterie etwa 5,2 kg. Insgesamt besitzt der Roboter 25 Freiheitsgrade, davon jeweils vier in jedem Arm, fünf in jedem Bein, zwei im Kopf, einen in der Hüfte, sowie einen im Handgelenk und einen für die Finger. Der im Kopf des Roboters eingebaute Computer ist mit einem energiesparenden Atom Z530 Prozessor mit 1,6 MHz Taktfrequenz ausgestattet. Es stehen insgesamt1 GB Arbeitsspeicher zu Verfügung. Das Betriebssystem ist ein Linux Open Embedded basierend auf Gentoo und befindet sich auf 2 GB internem Flashspeicher.

Der Computer besitzt zwei USB-Anschlüsse, eine serielle Schnittstelle und einen LAN Adapter. An einem der USB-Anschlüsse ist eine WLAN-Karte angebunden.

In der Brust des Roboters ist ein zweiter Prozessor eingebaut, der für die Steuerung der Motoren zuständig ist. Alle internen Sensoren des Roboters, mit Ausnahme der Kameras, sind ebenfalls an den Brustprozessor angeschlossen. Die Kom-

Software	Embedded GNU/Linux (basierend auf Gentoo)
Prozessor	Atom Z530 (Cache 512 kB, Takt 1,6 GHz, FSB 533 MHz)
Speicher	1 GB RAM, 2 GB Flash-Speicher, 8 GB micro sdhc
Architektur	x86
Netzwerk	WLAN IEEE 802.11b/g/n oder lan 1xRj45 10/100/1000 base T
Akku	lithium-ion 27,6 W h, 60 min Betriebsdauer
Dimensionen (HxTxB)	573 mm × 275 mm × 311 mm
Gewicht	5,2 kg
Freiheitsgrade	25 (Kopf 2, Hüfte 1, je Arm und Bein 5, je Hand 1)
Kamera	2 (Auflösung 1280 Pixel × 960 Pixel)
Bildeigenschaften	Format YUV422, Pixelgröße 1,9 µm, Signal-Rausch-Verhältnis 37 dB
Öffnungswinkel	72,6° diagonal und 60,9° horizontal
Fokus	30 cm bis unendlich
Interaktion	Knöpfe (Brust, 3x Kopf, 1 je Fuß), LEDs
PropriozeptiveSensoren	Hall-Sensoren für die Gelenkstellung (0,1° Präzision)
Kraftsensoren	FSR (Force Sensitive Resistors), 4 je Fuß
Inertialsensor	2 Gyrometer (eins pro Achse) Präzision 5 % 1 Beschleunigungssensor Präzision 1 %
Infrarot	Öffnungswinkel ±60°
Ultraschall	je 2 Empfänger und Sender Auflösung 1 cm und 60° Öffnung
Lautsprecher	2
Mikrofone	4 – Sensitivität 40 ± 3 dB

Tabelle 2.1: *Eine Übersicht der technischen Daten der Nao Roboterplattform (Modell H25). Eine vollständige Auflistung findet sich unter [Ald13].*

munikation mit dem Hauptprozessor erfolgt mit einem Takt von 10 ms über einen I^2C-Bus. Dabei werden die Kontrolldaten und die Sensordaten übertragen. Die Kontrolldaten, die an den Brustprozessor übertragen werden können, bestehen unter anderem aus Winkel- und Steifheitvorgaben für jedes Gelenk. Darüber hinaus werden Steuerdaten für die Abstandssensoren und LEDs übertragen. LEDs werden insbesondere zum *Debugging* verwendet. Das obere linke Bild der Abbildung 2.1 zeigt beispielsweise den Roboter, während sein Auge grün leuchtet. Im NaoTH-Framework bedeutet das, dass der Roboter glaubt einen Ball zu sehen.

Der Roboter ist zusätzlich ausgestattet mit vier Kraftsensoren (*engl. Force Resistive Sensors*, kurz: *FSR*) in jedem Fuß, einem Zwei-Achsen-Gyroskop, einem Drei-Achsen-Beschleunigungssensor und zwei Ultraschallsensoren in der Brust. Das Gyroskop und der Beschleunigungssensor sind dabei in der Nähe des Schwerpunktes eingebaut. Jedes Gelenk ist mit einem Winkelmesser ausgestattet, zusätzlich wird der Kontroll-Strom an jedem Gelenk gemessen.

Im Kopf des Roboters sind zwei vertikal angeordnete Kameras eingebaut, die als schwarze Punkte im Gesicht des Roboters zu erkennen sind. Die Kameras können parallel angesteuert werden und haben eine Auflösung von maximal 1280 x 960 Pixel, bei einer Bildwiederholrate von 30 fps (30 Bildern pro Sekunde). Der Roboter hat ein Gesichtsfeld, dass in der Diagonale 72,6° und in der Horizontalen 60,9° misst. Der Mensch hat dem gegenüber ein horizontales Gesichtsfeld von etwa 180°.

2.3 NaoTH-Framework

Im Rahmen der Teilnahme am RoboCup hat das Nao Team Humboldt (NaoTH) ein Framework entwickelt, das modular aufgebaut ist und sehr hardwarenah arbeitet. Die Hardwarenähe ermöglicht eine performante Ausführung der Algorithmen. So wird ermöglicht, dass der Roboter völlig autonom auf dem Spielfeld agieren kann.

Dieses Framework wird ebenso von den FUmanoids verwendet, einem RoboCup-Team der Freien Universität Berlin.

Die Debugging-Werkzeuge des Frameworks geben einen Einblick in den Roboter. Durch sie lassen sich die Richtigkeit der Modelle, des Verhaltens und der Perzepte analysieren. Diese Arbeit stützt sich zudem auf die bereits implementierte Bildverarbeitung und Odometrie-Berechnung. Sowohl die verwendeten Werkzeuge, als auch die Grundlagen und die Arbeitsweisen der implementierten Bildverarbeitung und Odometrieberechnung werden in folgenden Unterabschnitten näher beleuchtet.

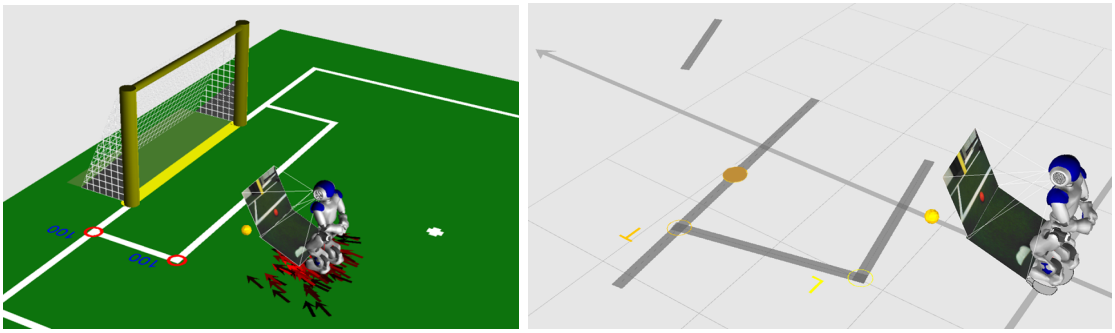

Abbildung 2.3: *Links: Die Kamerabilder des Roboters werden angezeigt. Unter ihm sind Hypothesen seines* Lokalisierungsfilters *zu erkennen. Im Bild sind die Randstücke der Linien zu erkennen, sowie ein Torpfosten. Rechts: Die Linien sind lokal in den Koordinaten des Roboters abgetragen.*

2.3.1 Vorhandene Werkzeuge

Das Framework ist modular aufgebaut. Module berechnen beispielsweise die Position des Balles oder die Position des Roboters. Die Berechnungen dieser Module werden in Repräsentationen gespeichert. Die Abbildung 2.3 zeigt einen Ausschnitt unseres Werkzeugs *RobotControl*. Damit sind wir in der Lage, Repräsentationen zu visualisieren. Auf einem Blick kann man die Kamerabilder und die Pose des Roboters erkennen. Die Projektion der Objekte im Bild (Linie, Torpfosten, Ball) auf das Spielfeld ist schön zu erkennen.

Mittels dieser Repräsentationen und Modelle kann der Roboter ein entsprechendes Verhalten ausführen. Das Verhalten besteht aus hierarchischen angeordneten endlichen Zustandsautomaten. Diese Automaten haben Zugriff auf Symbole der Repräsentationen. Als Programmiersprache für das Verhalten nutzen wir die *Extensible Agent Behavior Specification Language* (XABSL) [LBBJ04]. Die Abbildung 2.4 zeigt erneut das Werkzeug RobotControl. Diesmal ist das Verhalten des Roboters in einer bestimmten Spielszene dargestellt.

Im aktuellen Ausschnitt befindet sich der Roboter im Zustandsautomaten *move_around_ball* und führt die Aktion *turn_left* aus. Der Roboter dreht sich also links um den Ball. Dies kann nützlich sein, um sich in eine bessere Ausgangssituation für die nächste Aktion zu bringen. Überlicherweise sollte der Zustandsautomat durch das Ausführen der Aktion einen bestimmte neue Situation erzeugen und dadurch einen Endzustand erreichen. Denkbar wäre, dass der Roboter einen bestimmten günstigen Winkel zum Tor erreicht und die aktuelle Drehung um den Ball beendet.

Ausführliche Informationen zu den verwendeten Werkzeugen im NaoTH-Framework kann der Diplomarbeit von Florian Holzhauer [Hol13] entnommen werden.

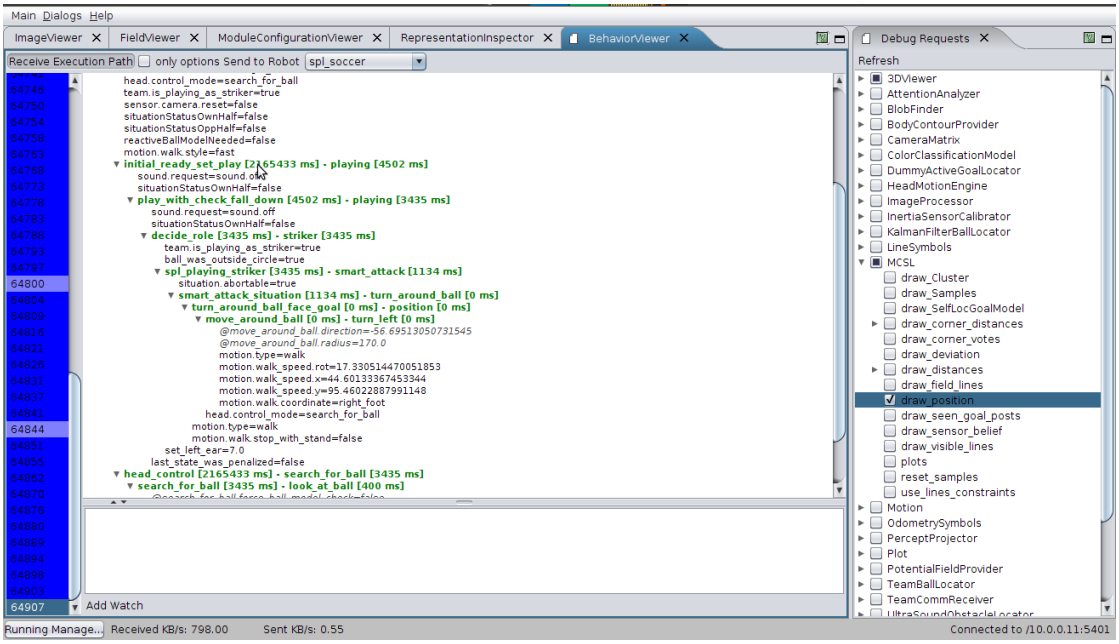

Abbildung 2.4: *Das Verhalten des Roboters in einem bestimmten Moment, dargestellt in unserem* RobotControl*-Werkzeug. Es zeigt die Situation, dass ein NAO sich um den Ball dreht und dabei versucht, sich zum Tor auszurichten.*

2.3.2 Kinematik und Odometrie

Dieser Abschnitt umreißt die Berechnung der Eigenbewegung des Roboters im NaoTH-Framework. Insbesondere hat die Bewegung Einfluss auf die Position der Kamera. Um beispielsweise die Entfernung zu einem Objekt im Kamerabild bestimmen zu können, ist dieses Wissen entscheidend.

In Abschnitt 2.2 wurden einige Sensoren des Naos genauer vorgestellt. Unter anderem besitzt der Roboter Positionssensoren in den Gelenken, Beschleunigungssensoren und Drucksensoren in den Füßen. Zusätzlich ist durch das offizielle Datenblatt [Ald13] die kinematische Struktur des Roboters bekannt, insbesondere die Position der Gelenke und die Länge der Gliedmaßen. Mit Hilfe dieser Informationen (Sensordaten und kinematische Struktur) kann ein kinematisches Modell des Roboters aufgebaut werden. Aus diesem Modell lassen sich die extrinsischen Parameter der Kamera ermitteln. Das heißt, man kann berechnen, wo sich die Kamera relativ zum Koordinatensystem des Roboters befindet und wie sie ausgerichtet ist.

Zusammen mit dem Wissen der intrinsischen Parameter – beispielsweise dem Öffnungswinkel – lässt sich daraus eine Kameramatrix errechnen. Mithilfe dieser Matrix lässt sich eine Zuordnung gesehener Objekte im Bild zu deren tatsächlicher Position in relativen Koordinaten des Roboters ermittelt. Detailliere Definition

und Diskussion zur Kameramatrix findet man in der Diplomarbeit von Heinrich Mellmann [Mel10, Abschnitt 2.1.4].

Berechnet man die Veränderung des kinematischen Modells über die Zeit, so lässt sich die Bewegung des Roboters ableiten. Insbesondere lässt sich einfach die Verschiebung der Füße relativ zueinander bestimmen. Zudem lässt sich anhand der Kraftsensoren in den Füßen gut schätzen, welcher Fuß sich gerade auf dem Boden befindet. Zusammen erlauben diese beiden Informationen den Rückschluss auf den Weg den der Roboter seit der letzten Messung zurückgelegt hat. Diese Messungen werden über die Zeit akkumuliert und als Odometrie-Modell zur Verfügung gestellt. Genauer gesagt gibt das Odometrie-Modell an weit sich der Roboter seit dem letzten Zeitschritt bewegt hat. In dieser Arbeit wird es durch eine zweidimensionale Transformation beschrieben, die die Translation und die Rotation des Roboters seit dem letzten Zeitschritt angibt. Weitere Informationen zum Odometrie-Bewegunsmodell findet man in [Mel10] und [TBF06].

Die Annahme, dass Motorkommandos exakt ausgeführt werden, ist nicht immer korrekt. Der Roboter könnte auf den Linien rutschen oder leicht am Untergrund durch Unebenheiten hängen bleiben. Darüber hinaus können andere Roboter oder Hindernisse zur Verschiebung führen [AH14]. Das vorgestellte Modell ist daher Fehlern unterworfen. Im Kapitel 7 wird eine Übersicht dieser Fehler empirisch ermittelt, sowie Verbesserungen des zugrunde gelegten Modells diskutiert.

2.3.3 Bildverarbeitung und Objekterkennung

Dieser Abschnitt stellt die Techniken zur Bildverarbeitung und Objekterkennung im NaoTH-Framework vor.

Im Abschnitt 2.1 wurde bereits darauf eingegangen, dass die Umgebung im RoboCup-Umfeld farbkodiert ist. Darüber hinaus entsprechen die vorhandenen Objekte zumeist exakten Maßen. Die Bildverarbeitung des NaoTH-Frameworks liefert so genannte Perzepte der bekannten Objekte dieser strukturierten Testumgebung. Ein Perzept besteht aus bestimmten Merkmalen, die ein Objekt charakterisieren. Zur Implementation eines Tormodells ist beispielsweise von größerer Bedeutung, Wissen zur Lage von Torpfosten zu verarbeiten. Die Position eines Pfostens bezüglich eines Roboters lässt sich beispielsweise aus der Distanz d und dem Winkel α bestimmen. Folgerichtig enthält ein Pfostenperzept $\mathcal{P}^t_{Pfosten}$ unter anderem diese Polarkoordinaten $(d, \alpha)_{Pfosten}$.

Das Einsparen von Ressourcen bei der Programmierung autonomer Roboter spielt eine wesentliche Rolle. Daher liegt dieser Arbeit eine Bildverarbeitung mittels spärlicher Daten zu Grunde (engl: *sparse image processing*). Zur Verarbeitung eines Bildes und Berechnung der Perzepte $\mathcal{P}$ wird nicht die maximal mögliche Auflösung verwendet. Das Bild wird anstatt dessen mit Scanlinien durchsucht. Abbildung 2.5 zeigt mit den oberen Bildern eine beispielhafte Szene aus der Roboter-

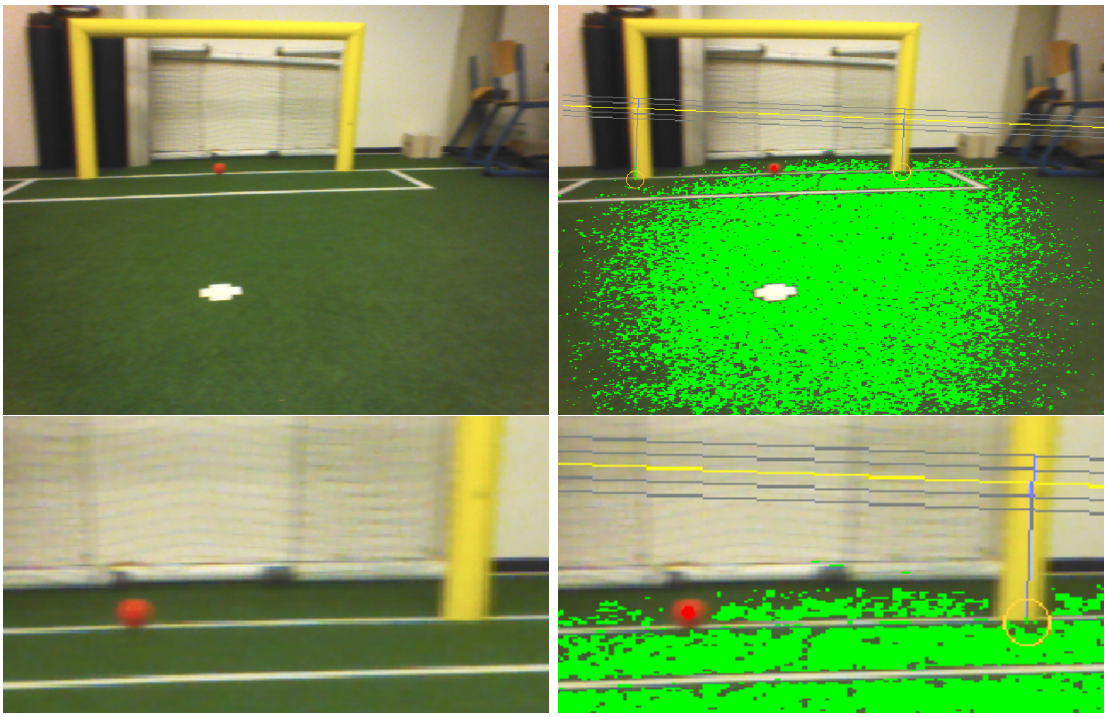

Abbildung 2.5: *Ausschnitt einer Szene des Experiments 6.1 aus Kapitel 6. Der Roboter steht dabei drei Meter vom Ball (zentriert zum Tor) entfernt.*

Sicht. Die unteren Bilder sind Vergrößerungen zum besseren Verständnis. Es sind fünf horizontale Scanlinien im oberen rechten Bild auszumachen. Im vorliegenden YUV422-Farbraum werden entlang dieser Linien Sprünge im V-U Differenzkanal detektiert. Gefundene Sprünge dienen der Bildverarbeitung dazu, um entlang mutmaßlicher Pfosten (querab der gefundenen Sprünge) zu scannen und entsprechende Fußpunkte der Pfosten zu ermitteln. Fußpunkte haben dabei die Farbe grün, so dass davon ausgegangen werden kann, dass man die Position des Pfosten-Perzepts $\mathcal{P}^t_{Pfosten}$ auf dem Feld ermitteln kann. Ist der Fußpunkt nicht grün, so kann davon ausgegangen werden, dass ein Roboter die Sicht auf den Pfosten verdeckt. In diesem Fall ist der relative Winkel α zum Pfosten zwar aussagekräftig, die Position des Pfostens auf dem Feld jedoch sehr wahrscheinlich nicht. Jedes Pfostenperzept $\mathcal{P}^t_{Ball}$ führt die Information zur Gültigkeit der Position mit sich. Diese visuelle Distanzbestimmung anhand von Referenzobjekten wird in der Diplomarbeit von Heinrich Mellmann [Mel10, Kapitel 2] detailliert beschrieben. Insbesondere wird der Einfluss von Fehlmessungen auf die Distanzschätzung diskutiert.

Ungenauigkeiten dieser Methode können an verschiedenen Stellen auftreten. Erste Anhaltspunkte sind in Abbildung 2.5 zu erkennen. Die Ausrichtung der Torpfosten zur Torlatte ist nicht rechtwinklig. Da dies aber in der Realität nahezu erreicht

wird, kann eine Verzerrung der Kamera ausgemacht werden. Dies entstammt zum einem der Kopfbewegung des Naos, zum anderem der sphärischen Form der Linse. Dieses Wissen muss in der Berechnung der Kameramatrix berücksichtigt werden, da andernfalls im Randbereich erhebliche Fehlmessungen zu erwarten sind.

Entlang vieler vertikaler Scanlinien wird ein Histogram über die vorhandenen Farben berechnet. Mit der Annahme, dass die Feldfarbe (grün) den größten Anteil stellt, ist diese Farbe dem Roboter bekannt.

Eine Einfärbung dieser detektierten Farbe sieht man in der Abbildung 2.5. Es ist zu erkennen, dass die Ausleuchtung des Bildes der sphärischen Form der Linse folgt. Dies ist anhand des kreisförmigen zentralen Grünabschnitts erkennbar. Die unterschiedliche Ausleuchtung führt zu unterschiedlichen Sprüngen entlang der Scanlinien. Dies wiederum führt dazu, dass Objekte in den verschiedenen Bildbereichen unterschiedlich gut erkannt werden können.

Im Kapitel 6 wird die dieser Arbeit zugrunde liegende Objekterkennung empirisch analysiert und erkannte Fehlerquellen diskutiert.

2.4 Ground Truth

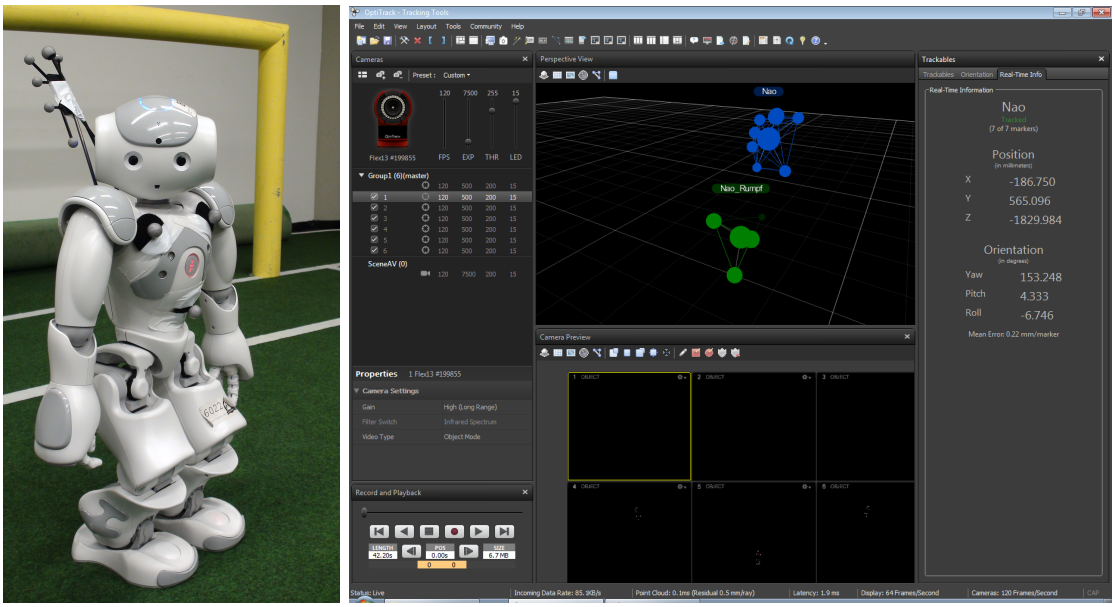

Abbildung 2.6: *Links: Der Nao ist mit zusätzlichen Markern ausgestattet. Diese sind an an seinem Rumpf befestigt, so kann die Rumpfbewegung des Roboters verfolgt werden. Rechts: Ein Ausschnitt aus den „Trackingtools“. Die oberen Marker (blau) und Rumpfmarker (grün) werden zu „Trackables“ zusammengefasst. Im rechten Bild erkennt man die aktuelle Pose des oberen Trackables im dreidimensionalen Raum.*

Das Trackingsystem besteht aus sechs Kameras *Flex 13*. Diese haben ein Öffnungswinkel von 56° und verarbeiten Daten bereits auf der Kamera. Die Auflösung beträgt maximal 1280 Pixel × 960 Pixel. Es werden 120 Bilder pro Sekunde verarbeitet [OpT14].

Um zu gewährleisten, dass der Roboter seinen Kopf bewegen kann, Objekte weiterhin wahrnimmt und die Körperrotation trotzdem anhand der Marker akkurat gemessen wird, sind die Marker am Rücken des Roboters befestigt. Zusätzlich befinden sie sich über den Kopf. So kann die Position des Roboters fast auf dem gesamten Feld gemessen werden. Die Genauigkeit je Marker betrug während der Experimente zwischen 0,4 und 1,3 mm je Marker.

3 Mathematische Grundlagen

Zur Implementation eines Tormodells ist es wichtig, dass der Roboter Informationen über die Torpfosten seiner Umgebung erhält. Darüber hinaus ist es nützlich, dass der Roboter eine Abschätzung seiner Eigenbewegung vorhält um die Position von Pfosten, die aktuell nicht im Sichtbereich des Roboters sind, geeignet vorherzusagen.

Im Abschnitt 2.3 werden grundlegende Informationen bereitgestellt, wie im Rahmen des NaoTH-Frameworks Positionsinformationen von Objekten und des Roboters bereitgestellt werden.

Die Kapitel 6 und 7 werden darüber hinaus empirisch die komplexen Fehlerverteilungen aufzeigen. Insbesondere wird die Erkenntnis folgen, dass die Fehler nicht normalverteilt sind. Daher kann im Rahmen dieser Arbeit nicht auf den sehr bekannten Kalmanfilter zurückgegriffen werden.

Um trotz verrauschter und unsicherer Daten eine valide Hypothese über den Standort eines Objekts im Raum zu bestimmen, bietet der Partikelfilter ein geeignetes Werkzeug um nicht-normalverteilte nicht-lineare Zustände zu modellieren [GSS93, IB96, SBFC03, VGP05].

Dieses Kapitel soll die Grundlagen und die Herleitung der Bayes'schen Modellierung in den ersten Abschnitten darlegen. Im letzten Abschnitt soll darüber hinaus mit dem Partikelfilter eine Approximation vorgestellt werden, die sich im Umfeld mobiler autonomer Roboter als nützlich erwiesen hat [PS99, SBFC03, VGP05].

Die Notation und Art der Herleitung ist maßgeblich geprägt durch das Buch *Probabilistic Robotics* [TBF06, Kapitel 1 u. 4] und die Vorlesung *Kognitive Robotik* [BH10, Abschnitt: Weltmodellierung].

3.1 Symbolübersicht

- x_t - Ein Zustand zum Zeitpunkt t (bspw. Torpfostenposition)
- u_t - Eine Aktion des Roboters zum Zeitpunkt t (bspw. Schritt ausführen)
- z_t - Beobachtung des Roboters zum Zeitpunkt t (bspw. Pfostensichtung)

- $P(x)$ - A-priori-Wahrscheinlichkeit
- $P(z|x)$ - Sensormodell
- $P(x|u, x')$ - Aktionsmodell
- $Bel(x)$ - A-posteriori-Wahrscheinlichkeit

3.2 Annahmen

Üblicherweise bietet die Infrastruktur typischer Roboterplattformen die Messung genau einer Beobachtung in einem festen Zeitschritt. Es kann also von einer Sequenz von Beobachtungen ausgegangen werden, die wie folgt notiert wird:

$$z_1, ..., z_4 = z_{t_1}, z_{t_2}, z_{t_3}, z_{t_4}, \text{ mit } t_u \leq t_v \text{ und } u \leq v \in \mathbb{N}.$$

Die Aktionen eines Roboters werden von einem *Pedometer* gemessen. Im Falle des Naos wird mithilfe von Vorwärtskinematik abgeschätzt, welchen Weg der Roboter zurückgelegt hat. Details dazu finden sich im Abschnitt 2.3.2. Die resultierenden Sequenzen folgen einer ähnlichen Notation:

$$u_1, ..., u_4, = u_{t_1}, u_{t_2}, u_{t_3}, u_{t_4}, \text{ mit } t_u \leq t_v \text{ und } u \leq v \in \mathbb{N}.$$

Obwohl die Torpfosten einen festen globalen Standort haben und somit statische Objekte sind, verändert sich ihre Position relativ zum Roboter in Abhängigkeit der Bewegung des Roboters. Aufgrund der Ausführungen von Aktionen u wird der Zustand eines Torpfostens verändert. Die daraus resultierende Sequenz von Zustanden wird wie folgt notiert:

$$x_1, ..., x_4, = x_{t_1}, x_{t_2}, x_{t_3}, x_{t_4}, \text{ mit } t_u \leq t_v \text{ und } u \leq v \in \mathbb{N}.$$

Die Abbildung 3.1 zeigt den zeitlich Verlauf dieser Sequenzen. Ein Torpfosten befindet sich in einem Zustand x_t. Durch eine Aktion u_t des Roboters verändert sich dieser Zustand. Bei akkurater Vorhersage der Aktion, kann ebenso vorhergesagt werden, wie sich die Beobachtung z_{t+1} des Roboters verändert.

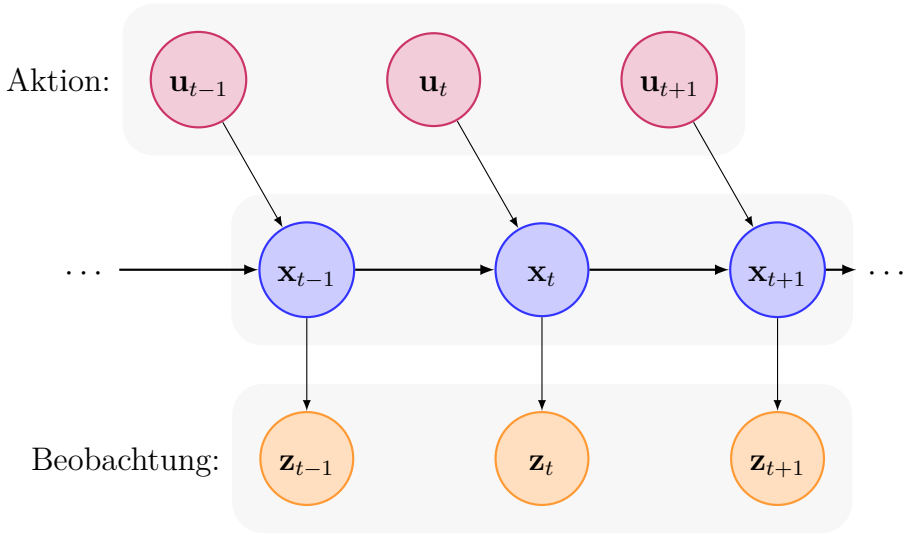

Abbildung 3.1: *Schematik eines Bayes'schen Netzes. Ein Torpfosten befindet sich in einem Zustand x_t. Durch eine Aktion u_{t+1} des Roboters verändert sich dieser Zustand. Bei akkurater Vorhersage der Aktion kann die Beobachtung z_{t+1} des Roboters vorhergesagt werden.*

Darüber hinaus wird angenommen, dass die Markow-Eigenschaft gilt. Diese Eigenschaft wurde vom Mathematiker Andrei Andrejewitsch Markow beschrieben und besagt, dass die Zustände x_{t-1} und x_{t+1} stochastisch unabhängig sind. Anders ausgedrückt bedeutet das, dass eine Beobachtung z_t zum Zeitpunkt t nur vom Zustand x_t abhängig ist, da alle vorhergehenden Informationen im aktuellen Zustand enthalten sind.

3.3 Bayes-Filter / Bayes'sche Modellierung

Die Bayes'sche Modellierung soll dazu verwendet werden um anhand von Beobachtungen des Roboters Aussagen zum Zustand eines Pfostens zu treffen.

Gesucht: Zum Zeitpunkt t ist die A-posteriori-Wahrscheinlichkeit des Zustands x_t nach den Beobachtungen $z_1, ..., z_t$ und der Ausführung der Aktionen $u_1, ..., u_t$ gesucht. Dieser unbekannten Umweltzustand nach der Beobachtung einer von x abhängigen Zufallsgröße, wird auch als *Belief* bezeichnet und üblicherweise mit *Bel* abgekürzt.

$$Bel(x_t) = P(x_t|z_1, ..., z_t, u_1, ..., u_t). \tag{3.1}$$

Dieser Zustand kann nicht direkt gemessen werden, ein Roboter muss für gewöhnlich von verschiedenen Daten auf den tatsächlichen Zustand schließen. Die Abschätzung eines Zustands und der dazugehörige tatsächliche Zustand wird so begrifflich getrennt.

Ebenfalls von Bedeutung ist der *Belief* im Zeitpunkt t vor der entsprechenden Messung der Beobachtung z_t. Diese entspricht einer *Zustandsvorhersage*.

$$\widehat{Bel}(x_t) = P(x_t|z_1, ..., z_{t-1}, u_1, ..., u_t). \tag{3.2}$$

Die Berechnung von $Bel(x_t)$ aus $\widehat{Bel}(x_t)$ wird auch *Korrektur* oder *Sensor-Update* genannt.

Gegeben: Folgende Formeln aus der Wahrscheinlichkeitstheorie müssen vorausgesetzt sein. Das **Theorem der totalen Wahrscheinlichkeit** besagt, dass unter der Bedingung, dass sich die einzelnen Ereignisse ausschließen, für ihre Summe folgende Formel gilt:

$$P(x|y) = \int P(x|y,z)\ P(z|y)\ \mathrm{d}z, \text{wobei} \int P(z)\ \mathrm{d}z = 1. \tag{3.3}$$

Im Rahmen der probabilistischen Robotik nimmt des Weiteren die **Bayes'sche Formel** eine fundamentale Bedeutung ein. Sie ist wie folgt beschrieben:

$$P(x|y) = \frac{P(y|x)P(x)}{P(y)}, \text{wobei } P(y) > 0. \tag{3.4}$$

Die angenommene Markow-Eigenschaft besagt, dass eine Beobachtung z_t zum Zeitpunkt t nur vom Zustand x_t abhängig ist, da alle vorhergehenden Informationen im aktuellen Zustand enthalten sind. Daraus ergeben sich folgende Formeln:

$$P(x_t|x_0, ..., x_{t-1}, z_1, .., z_{t-1}, u_1, .., u_t) = P(x_t|x_{t-1}, u_t), \tag{3.5}$$

$$P(z_t|x_0, ..., x_t, z_1, .., z_t, u_1, .., u_t) = P(z_t|x_t). \tag{3.6}$$

Das **Sensormodell** (auch: *Messungs-Wahrscheinlichkeit*) des Roboters modelliert die Beobachtungen die der Roboter durch seine Sensoren wahrnimmt. Dies kann beispielsweise die polaren Koordinaten, d.h. der Winkel und die Distanz eines Torpfosten sein.

$$P(z|x) \tag{3.7}$$

Des Weiteren ist das **Aktionsmodell** (auch: *Übergangs-Wahrscheinlichkeit*) gegeben.

$$P(x_t|x_{t-1}, u_t), \text{ mit } t \in \mathbb{N} \tag{3.8}$$

Das Sensormodell (3.7) und das Aktionsmodell (3.8) beschreiben zusammen das dynamische stochastische System eines Roboters und seiner Umgebung.

Ferner ist die **A-priori-Wahrscheinlichkeit** (oder: *Anfangswahrscheinlichkeit*) des Systemzustandes $P(x_0)$ gegeben.

Herleitung des Bayes-Filters: Zunächst nutzt man die Methodik der Induktion und beginnt mit der Anwendung der konditionellen *Bayes'schen Formel* entsprechend der Definition (3.4).

$$P(A|B,C) \cdot P(B|C) = P(B|A,C) \cdot P(A|C) \Leftrightarrow P(A|B,C) = \frac{P(B|A,C) \cdot P(A|C)}{P(B|C)}$$

Durch geeignetes Einsetzen der Beobachtungen z, Aktionen u und des Zustandes x aus Abschnitt 3.2 erhält man die nachfolgende Gleichung.

$$P(\overbrace{x_t}^{A} | \overbrace{z_1, ..., z_t, u_1, ..., u_t}^{B,C}) = \frac{P(\overbrace{z_t}^{B} | (\overbrace{x_t}^{A}, \overbrace{z_1, .., z_{t-1}, u_1, .., u_t}^{C}) P(\overbrace{x_t}^{A} | \overbrace{z_1, .., z_{t-1}, u_1, .., u_t}^{C})}{P(\underbrace{z_t}_{B} | \underbrace{z_1, .., z_{t-1}, u_1, .., u_t}_{C})}$$

$$Bel(x_t) \overset{(3.1)}{=} \frac{P(z_t|x_t, z_1, .., z_{t-1}, u_1, .., u_t) P(x_t|z_1, .., z_{t-1}, u_1, .., u_t)}{P(z_t|z_1, .., z_{t-1}, u_1, .., u_t)}$$

$$Bel(x_t) \overset{(3.6)}{=} \frac{P(z_t|x_t) P(x_t|z_1, .., z_{t-1}, u_1, .., u_t)}{P(z_t|z_1, .., z_{t-1}, u_1, .., u_t)}$$

Durch die Einführung eines Normierungsfaktors η für die Gesamtwahrscheinlichkeit 1, mit $\eta = \frac{1}{P(z_t|z_1,..,z_{t-1},u_1,..,u_t)}$ lässt sich die Gleichung übersichtlicher darstellen.

$$Bel(x_t) = \eta \cdot P(z_t|x_t) P(x_t|z_1, .., z_{t-1}, u_1, .., u_t)$$

$$\begin{aligned} Bel(x_t) &= \eta \cdot P(z_t|x_t) \overbrace{P(x_t|z_1, .., z_{t-1}, u_1, .., u_t)}^{\widehat{Bel}(x_t)} \\ Bel(x_t) &\overset{(3.2)}{=} \eta \cdot P(z_t|x_t) \ \widehat{Bel}(x_t) \end{aligned} \tag{3.9}$$

Gut ersichtlich ist in diesem Schritt, dass der eigentliche Zustand ein Produkt aus dem vorhergesagten Zustand und dem Sensormodell (*Messungs-Wahrscheinlichkeit*) ist.

Durch die Anwendung des Theorems der totalen Wahrscheinlichkeit und der Markow-Annahme für $\widehat{Bel}(x_t)$ ergibt sich folgende Gleichung.

$$\begin{aligned}\widehat{Bel}(x_t) &= P(x_t|z_1,..,z_{t-1},u_1,..,u_t) \\ &\overset{(3.3)}{=} \int P(x_t|x_{t-1},z_{1-1},..,z_t,u_1,..,u_t)\ P(x_{t-1}|z_1,...,z_{t-1},u_1,...,u_t)\ \mathrm{d}x_{t-1} \\ &\overset{(3.5)}{=} \int P(x_t|x_{t-1},u_t)\ P(x_{t-1}|z_1,...,z_{t-1},u_1,...,u_t)\ \mathrm{d}x_{t-1}\end{aligned}$$

Somit ergibt sich für $Bel(x_t)$ aus (3.9):

$$Bel(x_t) = \eta \cdot P(z_t|x_t) \int P(x_t|x_{t-1},u_t)\ P(x_{t-1}|z_1,...,z_{t-1},u_1,...,u_t)\ \mathrm{d}x_{t-1}.$$

Es ist aus der Gleichung zu erkennen, dass u_t aus den Bedingungen verworfen werden kann, da es keine Bedingung für den Zustand x_{t-1} darstellt, da er zeitlich nach diesem Zustand liegt. Durch das anwenden der A-posteriori-Wahrscheinlichkeit (3.1) ergibt sich folgende rekursive Gleichung:

$$\begin{aligned}Bel(x_t) &= \eta \cdot P(z_t|(x_t) \int P(x_t|x_{t-1},u_t)\ P(x_{t-1}|z_1,...,z_{t-1},u_1,...,u_{t-1})\ \mathrm{d}x_{t-1} \\ &= \eta \cdot P(z_t|(x_t) \int P(x_t|x_{t-1},u_t)\ \underbrace{P(x_{t-1}|z_1,...,z_{t-1},u_1,...,u_{t-1})}_{Bel(x_{t-1})}\ \mathrm{d}x_{t-1} \\ &\overset{(3.1)}{=} \eta \cdot P(z_t|(x_t) \int P(x_t|x_{t-1},u_t) Bel(x_{t-1})\ \mathrm{d}x_{t-1}.\end{aligned}$$

Die anschließende Formel kennzeichnet zum besseren Verständnis das enthaltene Sensor- beziehungsweise Aktionsmodell des nun hergeleiteten Bayes-Filters.

$$Bel(x_t) = \eta \cdot \underbrace{P(z_t|x_t)}_{Sensormodell} \int \underbrace{P(x_t|x_{t-1},u_t)}_{Aktionsmodell} Bel(x_{t-1})\ \mathrm{d}x_{t-1} \tag{3.10}$$

Es ist ersichtlich, dass zur Berechnung von $Bel(x_t)$ der Bayes-Filter Informationen über alle vorhergehenden Zuständen benötigt. Dies ist in einer Umwelt mit einer unendlichen Anzahl von Zuständen nicht umzusetzen. Aus diesem Grund gibt es eine Vielzahl an Approximationen des Bayes-Filters.

3.4 Partikelfilter

Partikelfilter werden gegenüber den weit verbreiteten Kalmanfiltern bevorzugt, sobald man den Zustand dynamischer nicht-linearer Prozesse abschätzen möchte. Ein Kalmanfilter zeichnet sich zwar durch die effiziente Berechnung einer Zustandsvorhersage aus, der zu messende Zustand muss allerdings einer unimodalen Normalverteilung entsprechen. Kapitel 6 und 7 werden zeigen, dass die Fehlerverteilung der Sensoreingaben nicht normalverteilt ist. Somit kann nicht davon ausgegangen werden, dass der abzuschätzende Zustand normalverteilt ist. Partikelfilter können nicht-lineare nicht-normalverteilte Prozesse abschätzen [GSS93, IB96, PS99, SBFC03, VGP05]. Für die Implementation des Multi-Hypothesen-Tormodells wurden daher Partikelfilter gewählt und werden im Folgenden vorgestellt.

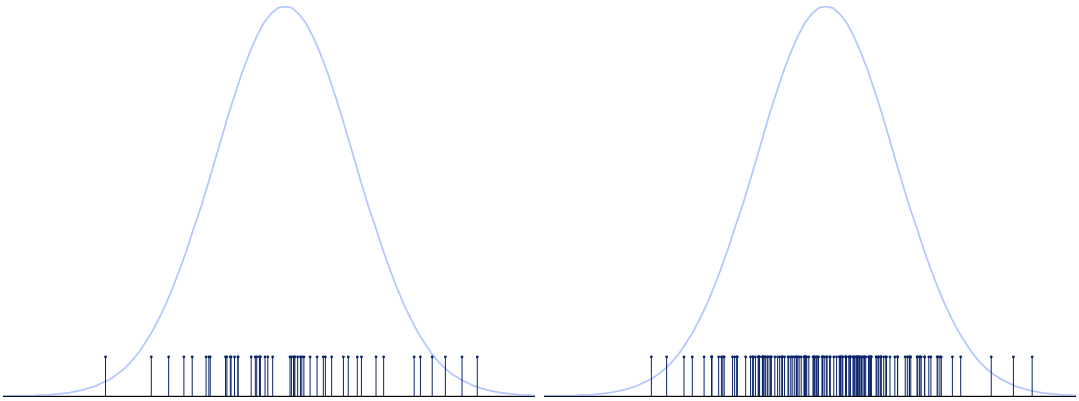

Abbildung 3.2: *Die Anzahl M der Partikel eines Partikelfilters ist sehr entscheidend. Einerseits muss die anzahl möglichst groß gewählt werden, um komplizierte Verteilungen eines Zustandes repräsentieren zu können. Andererseits erfährt jeder Partikel eine Neuberechnung anhand von Sensoreingaben. Dies führt mit großem M zu einem hohem Rechenaufwand. Die Abbildung links zeigt, wie eine Gaußfunktion durch die dichte von 50 Partikeln repräsentiert werden. Rechts, und deutlich akkurater, wurden 150 Partikel verwendet.*

Das Schlüsselkonzept eines Partikelfilters ist es, die A-posteriori-Wahrscheinlichkeit $Bel(x_t)$ (3.1) zum Zeitpunkt t durch eine Menge $\mathcal{S}_t$ von Partikeln anzunähern.

$$\mathcal{S}_t := s_t^{[1]}, s_t^{[2]}, ..., s_t^{[M]} \text{ mit } 1 \leq m \leq M$$

Jedem Partikel $s_t^{[m]}$ repräsentiert eine Hypothese des Zustandes des zu filternden Objekts. M beschreibt hier die Anzahl der verwendeten Partikel.

Da jeder Partikel in jedem Zeitschritt Zustandskorrekturen erfährt, ist die Anzahl der Partikel ausschlaggebend für den Konsum von Rechenkapazität. Für autonome

Roboter mit beschränkten Ressourcen ist M möglichst klein zu wählen. Andererseits muss M groß genug sein, um den gesuchten Zustand überhaupt darstellen zu können. Da die Zustandsverteilung zudem meist unbekannt ist, stellt dies eine Herausforderung dar.

Angenommen ein Zustand sei normalverteilt. Abbildung 3.2 zeigt, wie dieser Zustand mit 50 Partikeln (links) und 150 Partikeln (rechts) repräsentiert werden kann. Die Dichte der Partikel entspricht dabei dem Funktionswert. Man sieht bereits an diesem Beispiel, dass 150 Partikel die Gaußfunktion deutlich akkurater repräsentieren.

Um den Bayes-Filter zu approximieren, soll $Bel(x_t)$ durch die Menge der Partikel $\mathcal{S}_t$ ausgedrückt werden. Die Wahrscheinlichkeit eines Partikels der Menge $\mathcal{S}_t$ ist dabei proportional zum entsprechenden A-posteriori-Wahrscheinlichkeit des Zustandes (3.1):

$$s_t^{[m]} \sim P(x_t|z_1, ..., z_t, u_1, ..., u_t) = Bel(x_t)$$

Dieser Formel folgend, ist es umso wahrscheinlicher, dass der tatsächliche Objektzustand im Filter repräsentiert wird, je dichter die Partikel sind.

Entsprechend des Bayes-Filters (3.10) erzeugt der Partikelfilter die Menge von Partikeln $\mathcal{S}_t$ aus der vorhergehenden Menge $\mathcal{S}_{t-1}$, einem Aktions- und einem Sensormodell.

Algorithmus 1 : Partikelfilter mit *Importance Sampling*

Eingabe : $\mathcal{S}_{t-1}$, u_t, z_t
Daten : $M \leftarrow |\mathcal{S}_{t-1}|$, $\bar{\mathcal{S}} \leftarrow \emptyset$, $\mathcal{S}_t \leftarrow \emptyset$
Ausgabe : $\mathcal{S}_t$

1 **foreach** $m \in M$ **do**
2 | *Schätzung* $s_t^{[m]} \sim P(s_t|u_t, s_{t-1}^{[m]})$ (Aktionsmodell (3.8) anwenden)
3 | $w_t^{[m]} \leftarrow P(z_t|s_t^{[m]})$ (Sensormodell (3.7) anwenden)
4 | $\bar{\mathcal{S}} \leftarrow \bar{\mathcal{S}} + \langle s_t^{[m]}, w_t^{[m]} \rangle$
5 **end**
6 **foreach** $m \in M$ **do**
7 | Mit Wahrscheinlichkeit $\propto w_t^{[m]}$ ziehe $s_t^{[m]}$ (Tupel aus $\bar{\mathcal{S}}$)
8 | Füge $s_t^{[m]}$ zu $\mathcal{S}_t$ hinzu
9 **end**

Der Algorithmus 1 aus [TBF06, Seite 98] zeigt eine basale Variante eines Partikelfilters mittels *Importance Sampling*. Zunächst wird die Änderung des Partikels aufgrund der Aktion u_t geschätzt. Der Algorithmus gewichtet daraufhin jeden Partikel

der Menge $\mathcal{S}_{t-1}$ mit einem Gewicht w und speichert dieses Tupel in eine temporäre Menge $\bar{\mathcal{S}}$. Die Gewichtung wird dabei anhand der Beobachtung z_t berechnet. In einem nächsten Schritt werden Partikel aus $\bar{\mathcal{S}}$ gezogen und in $\mathcal{S}_t$ gespeichert. Je höher ihr Gewicht ist, desto wahrscheinlicher ist ihre Ziehung. $\mathcal{S}_t$ repräsentiert am Ende die A-posteriori-Wahrscheinlichkeit $Bel(x_t)$ (3.1) des Zustands x_t.

Zum besseren Verständnis werden im Folgenden die Schritte im Detail beschrieben:

1. Durch die Aktion u_t und dem Partikel $s_{t-1}^{[m]}$ wird in Zeile 2 des Algorithmus 1 eine Hypothese $s_t^{[m]}$ für den Zeitpunkt t erzeugt. Der Hochindex impliziert, dass die neue Hypothese durch den m-ten Partikel der Menge $\mathcal{X}_{t-1}$ generiert wurde. Um diesen Schritt ausführen zu können, muss das Aktionsmodell $P(x_t|x_{t-1}, u_t)$ (3.8) bekannt sein. Nach M Iterationen erhält man die Filterrepräsentation von $\widehat{Bel}(x_t)$ (3.2).

2. In Zeile 3 wird für jeden Partikel $s_t^{[m]}$ der *Wichtigkeitsfaktor* $w_t^{[m]}$ berechnet. Die Gewichtung wird dabei mittels des Sensormodells (3.7) berechnet. Anders ausgedrückt, je besser ein Partikel $s_t^{[m]}$ einer Beobachtung z_t bestätigt, desto höher ist sein Gewicht.

   Die Menge der gewichteten Partikel $\bar{\mathcal{S}}$ entspricht bereits der A-posteriori-Wahrscheinlichkeit $Bel(x_t)$ (3.1) des Zustands x_t.

3. Für die nachfolgende Berechnungen muss sichergestellt werden, dass die Verteilung der Partikel in $\mathcal{S}_t$ die Gewichtung der Partikel aus $\bar{\mathcal{S}}$ wiederspiegelt. Um dies zu ermöglichen wird ein *Resampling* durchgeführt.

   Zeile 6 bis 9 implementiert das so genannte *Importance Sampling*. Jeder Partikel $s_t^{[m]}$ wird dabei mit der Wahrscheinlichkeit $w_t^{[m]}$ gezogen und der Menge $\mathcal{S}_t$ hinzugefügt. Die Verteilung der Partikel $\bar{\mathcal{S}}$ entspricht nun der A-posteriori-Wahrscheinlichkeit $Bel(x_t)$ (3.1) des Zustands x_t. $\mathcal{S}_t$ wird in einem neuen Zeitschritt wieder die Grundlage einer Berechnung bilden.

   Partikel, die nicht in der neuen Menge $\mathcal{S}_t$ enthalten sind, wurden höchstwahrscheinlich geringer gewichtet. Potenziell relevante Partikel werden in die neue Menge übernommen. Die Gewichte werden nicht übernommen, einzig die Verteilung der Partikel repräsentiert die A-posteriori-Wahrscheinlichkeit des Zustands x_t ($Bel(x_t)$ (3.1)).

Es ist leicht ersichtlich, dass eine alternative Implementierung des Partikelfilters auf das Resampling verzichten könnte und stattdessen für jeden Partikel sein Gewicht multiplikativ speichert: $w_t^{[m]} = P(z_t|x_t^{[m]})\ w_{t-1}^{[m]}$. Auch diese Implemetation würde wie im zweiten Schritt die A-posteriori-Wahrscheinlichkeit des Zustands x_t

annähern. Je nach abzudeckender Umwelt bedarf diese Methode eine hohe Partikelanzahl. Partikel könnten sonst in Regionen geringer Relevanz verweilen.

Das Resampling verfolgt den Zweck, die Partikel in den relevanten Bereich zur Messung der A-posteriori-Wahrscheinlichkeit zu verschieben. Durch diese Technik kann ein Filter auch mit deutlich weniger Partikeln den Zustand eines Objekts abbilden.

Es gibt verschiedene Möglichkeiten, das Resampling zu implementieren.

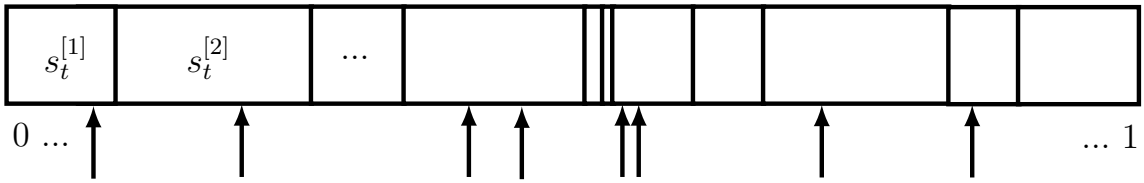

Abbildung 3.3: *Schematik der Importance-Sampling-Methode. Entlang eines Intervalls* $[0, ..., 1]$ *werden Partikel angeordnet. Die Größe eines Partikels ist durch seine Gewichtung determiniert. Daraufhin werden zufällig Zahlen aus dem Intervall gezogen (Pfeile). Je größer das Gewicht eines Partikels ist, desto wahrscheinlicher seine Ziehung.*

Importance Sampling wurde im Abschnitt 3.4 zur Umsetzung eines *Partikelfilters* verwendet und beschrieben. M Partikel der Menge $\bar{\mathcal{S}}$ werden anhand ihrer Gewichtungsfaktoren w gezogen oder nicht. Die Abbildung 3.3 visualisiert diesen Schritt.

Die Gewichtung bestimmt die *Ausdehnung* der Partikel $s_t^{[m]}$. In der Abbildung gilt somit $w_t^{[1]} < w_t^{[2]}$. Die Partikel spannen ein Intervall von 0 bis 1 auf. Entlang dieses Intervalls werden nun Zahlen gleichverteilt gezogen. In der Abbildung ist dies mit Pfeilen gekennzeichnet. Die gezogene Zahl ist einem Partikel zuordenbar. Partikel mit geringem *Gewichtungsfaktor* werden entsprechend seltener in die neue Menge von Hypothesen überführt.

Low Variance Sampling nutzt im Kern den selben Gedanken: Das Extrahieren vermeintlich *guter* Partikel. Allerdings geschieht das hier systematischer. Bei dieser Sampling-Methode wird lediglich eine Zufallszahl r gewählt und anhand dessen die weiteren Partikel nach der Überlegung $n = r + (m-1) \cdot M^{-1}$ gezogen, wobei $m \in [1, ..., M]$ gilt. Abbildung 3.4 verdeutlicht diese Überlegung.

Ein Vorteil besteht in der Systematik dieser Methode. Sollten alle Partikel gleiche Gewichtungsfaktoren aufweisen, so werden nach dieser Methode auch alle Hypothesen in die neue Menge überführt. Somit geht keine Hypothese verloren, auch wenn in einem Schritt keine Messung stattgefunden hat. Zudem verringert sich die Komplexität der Berechnung zu $O(M)$. Dies ist von entscheidender Bedeutung für die Notwendigkeit die Filter auf einem autonomen Roboter ausführen zu können.

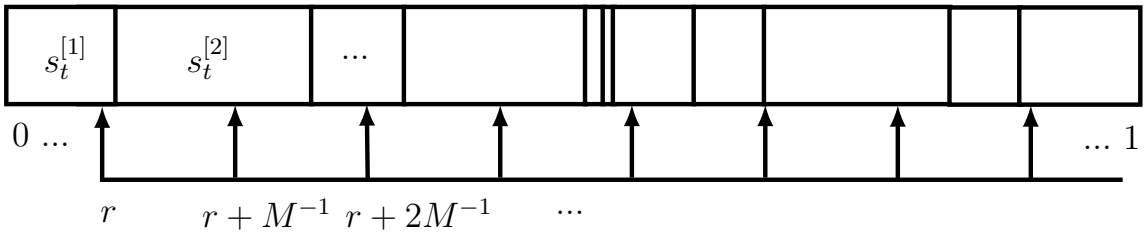

Abbildung 3.4: *Schematik der Low-Variance-Sampling-Methode. Die Partikel werden genau wie bei der Importance-Sampling-Methode angeordnet. Die Ziehung der Partikel geschieht im Unterschied dazu allerdings nicht unabhängig.*

Zur Vervollständigung des Algorithmus 1 zeigt der nachfolgende Algorithmus 2 aus [TBF06, Seite 110] die Methode im bekannten Pseudocode.

Algorithmus 2 : Low Variance Sampler

Eingabe : $\mathcal{X}_t$, $\mathcal{W}_t$
Daten : $\bar{\mathcal{X}} \leftarrow \emptyset$, $M \leftarrow |\mathcal{X}_t|$,
$r \leftarrow rand(0, M^{-1})$,
$c \leftarrow w_t^{[1]}$, $i \leftarrow 1$
Ausgabe : $\bar{\mathcal{X}}_t$

```
for m = 1 .. M do
    U ← r + (m − 1) · M^{-1}
    while U > c do
        i ← i + 1
        c ← c + w_t^[i]
    end
    Füge x_t^[i] zu X̄_t hinzu
end
```

4 Verwandte Arbeiten

Viele RoboCup-Mannschaften nutzen zur Verhaltenssteuerung die globale Feldposition aus [RLM+13, CKU+10, MSBH14]. Alle zur Verfügung stehenden Landmarken (bspw. Pfosten und Linien) werden dabei aggregiert. *Guerrero* und *Ruiz-del-Solar* schlugen 2007 vor, temporäre Landmarken in die Lokalisation mit einzubeziehen. Sie nutzen dafür Informationen, die sie als räumlich und zeitlich *lokale* Informationen bezeichnen. Das Nutzen dieser lokalen Informationen soll helfen, den Odometrie-Fehler abzuschätzen und somit die globale Lokalisierung zu verbessern [GS08]. Die Veröffentlichung greift ebenso das Thema auf, dass Fußballspieler zum Ausfüllen ihrer Rolle während eines Spiels keine exakte Position benötigen. *Tasse* beschrieb in seiner Dissertation [Tas14] eine Multi-Hypothesen-Ansatz zur Lokalisierung auf dem Feld. Hierbei wird der Ansatz verfolgt, dass bei abweichenden Sensorwahrnehmungen neue Hypothesen erzeugt und verfolgt werden.

Innerhalb der *Computer Vision* kommt dem *Tracking* einer Mehrzahl von Objekten ein bestärktes Interesse zu. Eine Vielzahl an Publikationen bezüglich der Schlagworte *Multi-Target Tracking* oder *Multi-Object Tracking* bestätigen dies [YMC07]. Diese Entwicklung liegt begründet im ungebrochenen Interesse an zuverlässigen Überwachungssystemen [Sit64, Rei79, YMC07, BBM09]. Neben militärischem Interesse [Sit64], spielt das Erfassen und Verwerten von Radar-Daten bei der Flugzeugüberwachung eine wichtige Rolle [Rei79], aber auch das Überwachen von Fußgängerzonen [YMC07] oder das Messen des Verkehrsflusses.

In diesem Kapitel werden die Anfänge und grundlegenden Methodiken für Modellierungsaufgaben beleuchtet, die hauptsächlich durch die Radar-Technik geprägt waren. Im Zweiten Abschnitt werden moderne Ansätze vorgestellt, die unter anderem durch die verbesserte visuelle Sensorik neue Möglichkeiten und Methoden aufbrachten. Abschließend wird die Relevanz der untersuchten Forschungsergebnisse in Bezug auf einen möglichen Lösungsansatz des Untersuchungsschwerpuntkes diskutiert.

4.1 Anfänge und Grundlagen

Bereits 1964 veröffentlichte *Robert W. Sittler* einen militärisch inspirierten Algorithmus, um verschiedene Zielobjekte mit einem Radar zu verfolgen [Sit64]. *Donald Reid* baute auf diese Arbeit auf und publizierte 1979 einen Implementati-

onsvorschlag zum Verfolgen (*tracking*) mehrerer Zielobjekte (*multi-target*) in einer ungeordneten Umgebung [Rei79], wobei er Radarsensoren verwendete. Viele nachfolgende und auch moderne Werke [YMC07] beziehen sich auf die Veröffentlichung *An Algorithm for Tracking Multiple Targets*. *Reid* nennt die aus dem Algorithmus resultierenden Hypothesen bezüglich eines Zielobjekts *Tracks*. Messungen mit vorhandenen *Tracks* zu assoziieren addressiert er als wichtigstes Problem des Verfolgens mehrere Zielobjekte (*multi-target tracking*) und stellt in der Veröffentlichung einen Implementationsvorschlag vor. Er konkretisierte die Probleme des Filterns mehrerer unmarkierter Messergebnisse mit folgenden Punkten:

1. Erhalten möglicher Messergebnisse, die in Wirklichkeit keinem Zielobjekt entsprechen (*false-positives*)
2. Unbekannte Zielobjekte erfordern das Initialisieren neuer *Tracks*
3. Vorhandene Zielobjekte werden nicht erkannt (bspw. durch Überdeckung)
4. Zielobjekte können verschwinden

Anders als *Sittler* setzt *Reid* in seiner Implementation nicht den Fall um, dass Zielobjekte ganz und gar verschwinden [Rei79]. Die Hervorhebung dieses Punktes (4) durch *Sittler* liegt im militärischen Blickwinkel seiner Arbeit begründet [Sit64].

Bereits aus der oben genannten Aufzählung wird ersichtlich, dass eine besondere Schwierigkeit darin besteht, den Zustand von Punkt 1 und Punkt 2 zu trennen. Wann handelt es sich hier um eine falsche Messung und wann um ein neues Zielobjekt? Eine ähnliche Problematik ist bezüglich Punkt 3 und 4 auszumachen. Wann gilt ein Objekt als verschwunden oder verdeckt?

Reid bezeichnet die wesentlichen Bestandteile seines *Brute-Force-Algorithmus'* als Cluster, Messungen und Hypothesen. Messungen werden mit vorhandenen Clustern assoziiert. So sie zu keinem Cluster passen, werden neue Cluster initialisiert. Eine Hypothese assoziiert eine Messung mit jedem vorhandenen Cluster. Für jede Messung wird eine Hypothese berechnet. Der Algorithmus behält in jedem Zeitschritt aktuelle Hypothesen und die Hypothesen der vorangegangenen N-Zeitschritte bei und aktualisiert diese mit neuen Sensor-Eingaben. Der Speicher- und Rechenaufwand steigt exponentiell mit den erhaltenen Messungen. Um diese Aufwendung einzuschränken, werden Hypothesen mit verschiedenen Mitteln gelöscht. Mittels eines Kalmanfilters [Kal60] berechnet der Algorithmus die entsprechende Abschätzung, zu welchem Ziel-Objekt eine Hypothese gehört. Nur wenn die Wahrscheinlichkeit einen bestimmten Grenzwert übersteigt, bleibt diese Hypothese Gegenstand der Untersuchung.

Der vorgestellte Algorithmus kann mit nicht-linearen Messungen und dynamischen Prozessen nicht umgehen. Zielobjekte die abrupt ihre Richtung ändern, werden

also nicht erkannt. Eine maßgebliche Kritik durch *Reid* selbst ist, dass der Algorithmus nicht adaptiv genug ist. Die Vorhersage der Zielobjekte unterliegt einheitlichem statistischen Wissen [Rei79].

Die Veröffentlichung von *Bar-Shalom* und *Tse Tracking in a Cluttered Environment with Probabilistic Data Association* von 1975 erweitert die Methoden zur Assoziation von Messungen und Vorwissen um ein Bayes'sches Verfahren. Die Autoren nennen ihr Verfahren *probabilistische Daten-Assoziation* (engl. *Probabilistic Data Association*, kurz: PDA). Die zugrunde liegende Annahme ist, dass jedes Objekt maximal eine Messung auslöst. Alle weiteren Messungen werden als Störungen modelliert. Der neue Zustand des Zielobjekts wird bedingt durch die Menge aller Messungen berechnet [BST75]. *Fortmann*, *Bar-Shalom* und *Scheffe* erweiterten in ihrer Publikation von 1983 *Sonar Tracking of Multiple Targets using Joint Probabilistic Data Association* diesen Ansatz um die Möglichkeit mehrere Objekte zu verfolgen [FBSS83]. Das Verfahren der *gemeinsamen probabilistischen Daten-Assoziation* (engl. *Joint Probabilistic Data Association*, kurz: JPDA) unterliegt der Annahme, dass eine Messung genau einem verfolgten Zielobjekt zuzuordnen ist und die Anzahl der zu verfolgenden Zielobjekte bekannt ist [Cox93]. Das Verfahren modelliert so die Assoziationsunsicherheit (Assoziation der Messung mit einem Zielobjekt) mithilfe einer Bayes'schen Abschätzung. Um die Zustände der Zielobjekte vorherzusagen, werden auch in diesem Ansatz Kalmanfilter angewandt.

4.2 Aktuelle Arbeiten

In einer Reihe von Veröffentlichungen setzten sich *Schulz*, *Burgard*, *Fox* und *Cremers* mit dem Verfolgen von mehreren sich bewegenden Objekten durch einen mobilen Roboter (*Tracking Multiple Moving Targets with a Mobile Robot*) auseinander [SBFC01b, SBFC01a, SBFC03]. Die Veröffentlichungen bieten einen Implementationsvorschlag zur Bewegungsanpassung eines mobilen Roboters in der realen (somit auch dynamischen) Welt. Mehrere sich bewegende Objekte werden von einem mobilen Roboter mit Laser-basierten Entfernungsmessern detektiert und verfolgt (*Tracking*). Die Sensoren erfassen mit 1° Genauigkeit 360° der Umwelt des Roboters, ohne sich zu überlagern. Das Bewegungsmodell der beobachtbaren Objekte ist die Lösungsgrundlage für die entscheidende Positionsmessung umgebender Objekte [SBFC01b].

Ein denkbares Anwendungsszenario lässt einen mobilen Roboter seine Bewegung anhand der Geschwindigkeit der umgebenden Menschen anpassen, um eine möglichst unauffällige Integration in die Umgebung zu ermöglichen. Die Kollisionsdetektion des Roboters kann überdies verbessert werden, indem sich kreuzende Trajektorien von Menschen und Roboter vorausberechnet werden [SBFC03]. Die

Arbeit stützt sich auf eine Variante des bereits vorgestellten JPDA-Filters (vgl. 4.1) und zeigt, wie das JPDAF-Framework mit *Partikelfiltern* [GSS93, PS99] anstelle des Kalmanfilters umgesetzt werden kann. Ein neuartiger Gedanken, denn bis zum Jahr 2001 nutzten nahezu alle Ansätze, die mehrere Objekte *tracken*, Kalmanfilter [SBFC01b].

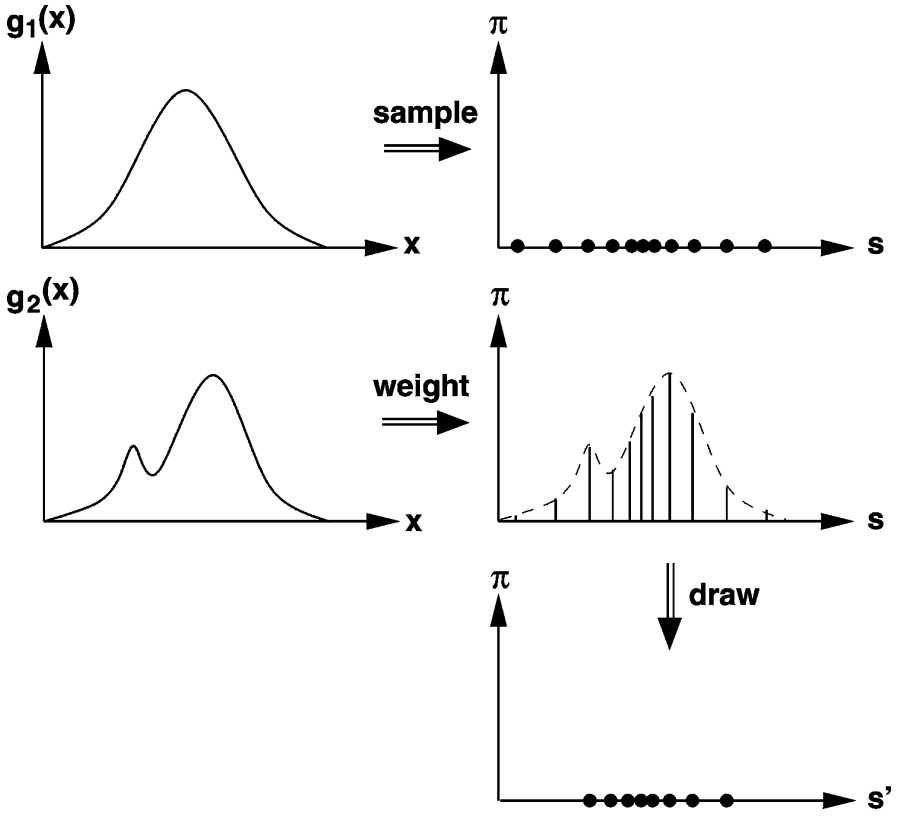

Abbildung 4.1: *Schema zum Partikelfilter aus [KMA01]. Oben: Eine Wahrscheinlichkeitsverteilungsfunktion $g_1(x)$ kann von einer Menge von Partikeln $s_1^t, ..., s_N^t$ repräsentiert werden. Darunter: Eine Beobachtung mit der Fehlerfunktion $g_2(x)$ trifft ein. Gewichte $\pi_1^t, ..., \pi_N^t$ werden für jeden korrespondierenden Partikel ermittelt. Entsprächend der Gewichte werden die Partikel neu gezogen (draw) und der Filter repräsentiert einen neuen Zustand. Abschnitt 3.4 formalisierte den Partikelfilter.*

Der Partikelfilter wurde von *Gordon*, *Salmond* und *Smith* 1993 unter dem Namen *Bootstrapfilter* eingeführt [GSS93] und wurde in der *Computer Vision* als *CONDENSATION Algorithmus* [IB96, KMA01] bekannt. Die Abbildung 4.1 verdeutlicht das Grundschema eines Partikelfilters. Der Filter wird zu einem Zeitpunkt t von einer Menge an Partikeln $s_1^t, ..., s_N^t$ repräsentiert. Diese Partikel sind gewichtet mit $\pi_1^t, ..., \pi_N^t$. Die Graphik oben rechts der Abbildung 4.1 zeigt eine solche Menge für den eindimensionalen Fall. Diese Menge soll die Approximation der gauß'schen

Wahrscheinlichkeitsverteilungsfunktion $g_1(x)$ darstellen und repräsentiert den aktuellen unsicheren Zustand x, in dem sich ein zu filterndes Objekt gerade befindet.

Im nächsten Zeitschritt t_1 trifft eine neue Beobachtung ein. Beobachtungen sind für gewöhnlich mit Fehlern behaftet. Die Fehlerverteilung einer Beobachtung lässt sich beispielsweise experimentell ermitteln und so in einem Beobachtungsmodell darstellen. Die Funktion $g_2(x)$ entspricht einem solchen Modell. Jedem Partikel des vorhergehenden Zeitschritts lässt sich ein Gewicht entsprechend dem Beobachtungsmodell zuweisen. Der eigentliche *Trick* [TBF06] eines Partikelfilters geschieht danach und wird in der Abbildung als Ziehung (engl: *draw*) beschrieben.

Entsprechend der Gewichte werden Partikel *gezogen*. Dies kann je nach Gewichtung mehrfach passieren. Manche Partikel mit geringer Relevanz erfahren keine Berücksichtigung und verschwinden dadurch. Aufgrund dieser Neuziehung kann eine neue Zustandsverteilung entstehen. Eine formale und ausgiebige Beschreibung erfolgte bereits im Abschnitt 3.4.

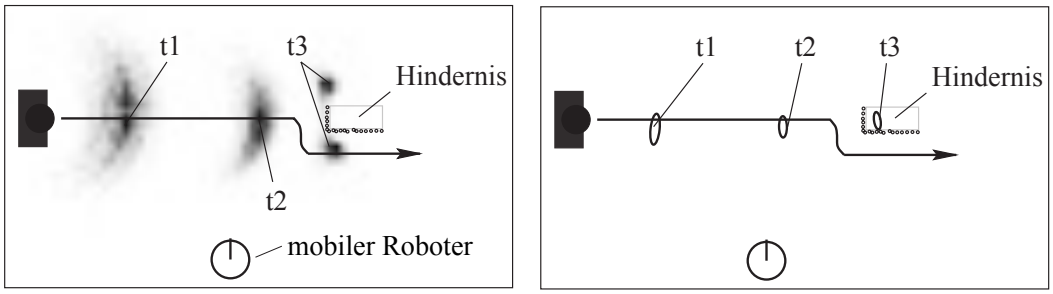

Abbildung 4.2: *Ein Experiment aus [SBFC03]. Ein mobiler Roboter erkennt eine Person (Zielobjekt) und verfolgt diese. Die Laufrichtung des Menschen ist durch den Pfeil markiert. Beide Graphiken zeigen die Zukunftsabschätzung bezüglich des Zustandes x zum Zeitpunkt t der Person. Links: Die Vorhersage eines Zustands wird mittels eines Partikelfilters berechnet. Die Partikel $s_1^t, ..., s_N^t$ sind als graue Punkte repräsentiert. Je dunkler diese sind, desto höher ist ihre Relevanz. Die Partikel repräsentieren die Wahrscheinlichkeitsverteilungsfunktion des Zustands x_t. Rechts: Der zukünftige Zustand der Person wird mit einem Kalmanfilter abgeschätzt. Der Filter kann mit der nicht-linearen Zustandsänderung der Person nicht umgehen [SBFC03]. Die Abschätzung ist beim dritten Zeitschritt (Im Bild t3) dadurch in einem Hindernis.*

In [SBFC03] wird die Verwendung des Partikelfilters motiviert. Ein Kalmanfilter, wie er bei den bisher vorgestellten Algorithmen Verwendung findet [Sit64, Rei79, FBSS83], zeichnet sich durch die effiziente Berechnung einer Zustandsvorhersage aus. Dennoch ist er *nur* optimal, wenn der zu messende Zustand einer unimodalen Normalverteilung entspricht. Partikelfilter hingegen können nicht-lineare nichtnormalverteilte Prozesse abschätzen [GSS93, IB96, PS99, SBFC03, VGP05]. Die Abbildung 4.2 aus [SBFC03] verdeutlicht den Unterschied. In dem Experiment

soll von einem Roboter ein Mensch *getracked* werden. Der Mensch ist als dunkle Box, der Roboter als schwarz umrandeter Kreis dargestellt. Ein Pfeil markiert die Laufrichtung des Menschen. In verschiedenen Zeitschritten t_n schätzt der zu Grunde liegende Filter die Position des Menschen zum Zeitpunkt t_{n+1} ab. Dies dient der Messungen des Zeitschrittes t_{n+1} mit dem erwarteten Zustand des Menschen assoziieren zu können. Das linke Bild zeigt die Flexibiltät eines Partikelfilters, der ein Hindernis in die Berechnung mit einbezieht. Das rechte Bild der Abbildung 4.2 zeigt das gleiche Experiment mit einem Kalmanfilter. Die Vorhersage dieses Filters für den Zeitschritt t_3 ist aufgrund der nicht-linearen Zustandsänderung, dass die Person sich im Hindernis befindet.

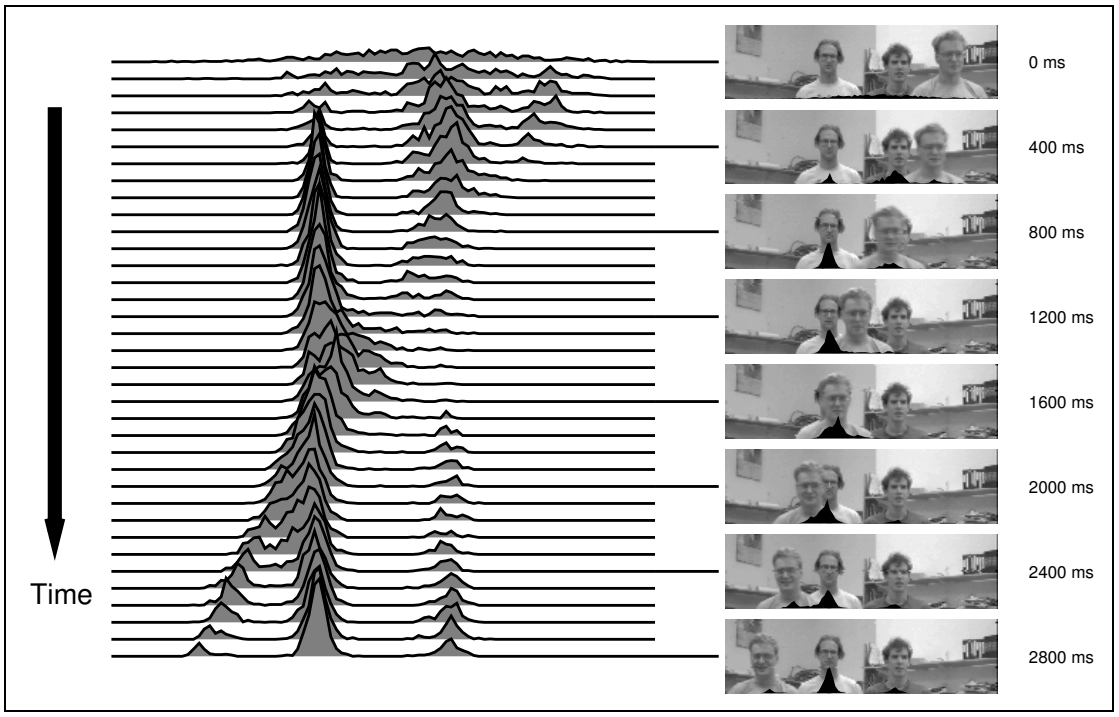

Abbildung 4.3: *Versuch aus [IB96]. Drei Gesichter sind zu sehen die ihre Position wechseln und sich dabei teilweise überdecken. Korrespondierende Wahrscheinlichkeitsverteilungsfunktionen sind entsprechend dargestellt. Es ist zu erkennen, dass sich die Partikel bezüglich der Gesichtsposition bereits nach* 400 ms *häufen. Trotz der Bewegung der Personen bleiben diese Häufungen weitestgehend erhalten. Es ist aber auch zu erkennen, dass die Häufungen unterschiedlich ausgeprägt sind.*

Um diese Problematik des Kalmanfilters zu umgehen, wurde der bereits vorgestellte JPDA-Filter von verschiedenen Arbeitsgruppen adaptiert und für die Verwendung eines Partikelfilters angepasst [SBFC03, VGP05]. Der für Partikelfilter erweiterte Algorithmus der JPDA-Filter wird von den Autoren *Schulz*, *Burgard*, *Fox*

und *Cremers* SJPDAF genannt [SBFC01b, SBFC01a, SBFC03]. Das S steht dabei für *Sample-Based*. Der SJPDA-Filter besteht aus einer festen Sample-Menge. Ein Gewichtungsfaktor β_{ji} berechnet die Wahrscheinlichkeit, dass eine Messung j zu einem Objekt i gehört. Beim *draw* der Abbildung 4.1 wird diese Gewichtung β zusätzlich mit einfließen. Somit kann sichergestellt werden, dass Sample nur von einer korrespondierenden Messung *verändert* werden. Das Problem, dass der Rechenaufwand mit jeder Beobachtung kombinatorisch wächst wird mit der Technik *Gating* angegangen, ähnlich wie es [Rei79] löste. Im SJPDA bereits im Vorfeld vernachlässigbare Wahrscheinlichkeiten nicht berücksichtig [FBSS83, SBFC03, VGP05]. Zusätzlich werden verdeckte Zielobjekte separat behandelt. Sind Sample nach der Berechnung seiner Vorhersage im überdeckten Bereich einer anderen Person, werden diese nicht mit Messungen assoziiert. Dies löst das Problem der Überdeckungen für *kurze* Phasen sehr elegant und führt auch zu einer Verbesserung des Trackings von Objekten trotz Überdeckungen [SBFC03].

Andere Ansätze nutzten den Partikelfilter ohne eine explizite Zuordnung der Beobachtung zu einem Partikel [IB96, KMA01], jede Beobachtung verändert demnach alle Partikel. Hierbei werden die erzeugten Partikel mit allen Beobachtungen gewichtet, und die gemeinsame Verteilung nach Häufungen durchsucht. *Isard* und *Blake* stellen in ihrer Veröffentlichung *Contour tracking by stochastic propagation of conditional density* einen solchen Filter mit gemeinsamer Wahrscheinlichkeitsverteilung vor. Die Abbildung 4.3 zeigt die Funktionsweise. Zur Vereinfachung ist das Problem nur eindimensional dargestellt. Drei Gesichter sind zu sehen die ihre Position wechseln und sich dabei teilweise überdecken. Unter den Gesichtern und separat links im Bild ist die resultierende Verteilung dargestellt. Es ist zu erkennen, dass sich die Samples bezüglich der Gesichsposition bereits nach 400 ms häufen. Trotz der Bewegung der Personen bleiben diese Häufungen erhalten.

Die Verwendung eines reinen Partikelfilters hat den Vorteil, dass keine kombinatorisch exponentiellen Assoziationen von Messungen bzgl. Zielobjekt/Partikeln wie in [FBSS83, SBFC03, VGP05] existieren. Es lässt sich jedoch schnell erkennen, dass der Einsatz einer multimodalen Verteilung [IB96, KMA01] zur Repräsentation aller Objekte nur gut funktioniert, wenn in jedem Zeitschritt auch wirklich alle Objekte gesehen werden. Ansonsten *wandern* die Partikel zu einem Punkt, andem die Messungen ihnen besser entsprechen [SBFC03] (vgl. 3.4).

Abbildung 4.4 aus [SBFC03] fasst dies zusammen. Die obere Bildfolge zeigt, wie ein Partikelflter mit nur einer gemeinsamen Menge von Partikeln [IB96, KMA01] auf ein erkanntes Hindernis (*obstacle*) reagiert. Die Partikel die hinter einem Bereich fallen, wo mögliche Messungen nicht vollzogen werden können, *driften* zum Ort der bestätigten Messungen. Die untere Bildfolge zeigt den Ansatz, dass Sample nur verändert werden, wenn sie mit einer Messung assoziiert werden. Partikel in

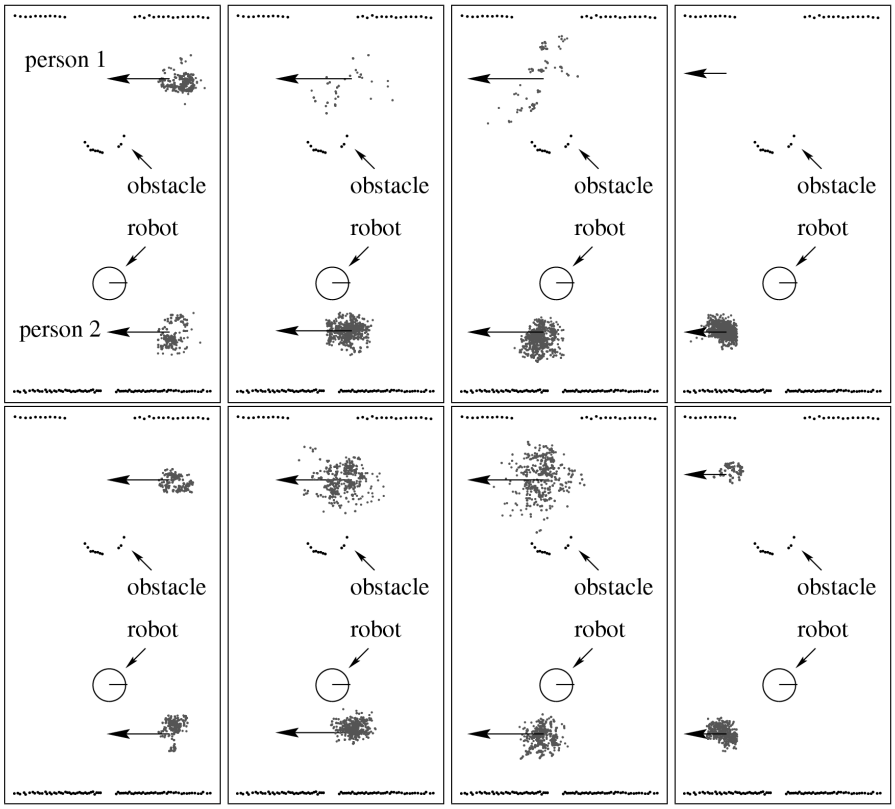

Abbildung 4.4: *Die obere Bildfolge zeigt das verhalten eines Partikelfilters mit nur einem gemeinsamen Menge von Samplen wie in [IB96, KMA01] auf. Sample* wandern *zum Bereich der durch Messungen bestätigt wird. Die untere Bildfolge zeigt den Ansatz für das nutzen von assoziierten Partikel-Mengen. Trotz steigender Unsicherheit (Partikel „driften“ auseinander) werden sie vom „Driften“ gehindert, da die ihre Überdeckung aktiv berücksichtigt wird [SBFC03].*

überdeckten Bereichen werden gesondert behandelt. Trotz steigender *Unsicherheit* (Partikel *driften* auseinander) bleibt eine Verfolgung möglich [SBFC03].

Im Ansatz zur Repräsentation mehrerer Objekte mit einer Verteilung [IB96, KMA01] können sich die Häufungen stark unterscheiden, wenn ein Zielobjekt beispielsweise permanent nur schlecht erkannt wird. Diese Diskrepanz kann verschiedene Objekte oder auch schlichtweg einen Fehler darstellen [KMA01].

Bei Erweiterung des Lösungsvorhabens ist daher zusätzlicher Assoziationsaufwand auch in [IB96, KMA01] nötig, da nicht zwingend klar ist, ob eine Häufung zu entsprechenden Zielobjekten oder Fehlmessungen korrespondiert. In [MB00] wird dies mit einer Partitionierung der Partikel-Menge getan. Ein gutes Einsatzszenario für einen reine Partikelfilter mit einer Wahrscheinlichkeitsverteilung zeigen *Koller-Meier* und *Ade* in *Tracking multiple objects using the Condensation algorithm* auf.

Ein mobiler Roboter weicht dabei den erkannten stationären Hindernissen aus. Da Verdeckungen von Hindernissen nur von anderen Hindernissen erzeugt werden können, ist ein Ausweichen bei bestimmten Häufungen immer sinnvoll. Komplizierte Trajektorien bestimmter Hindernisse sind nicht vorhersagbar. Es ist zur Lösung dieses Szenarios auch nicht elementar, um welches Hindernis es sich explizit handelt [KMA01]. Das *Tracking-Problem* wird somit nicht gelöst.

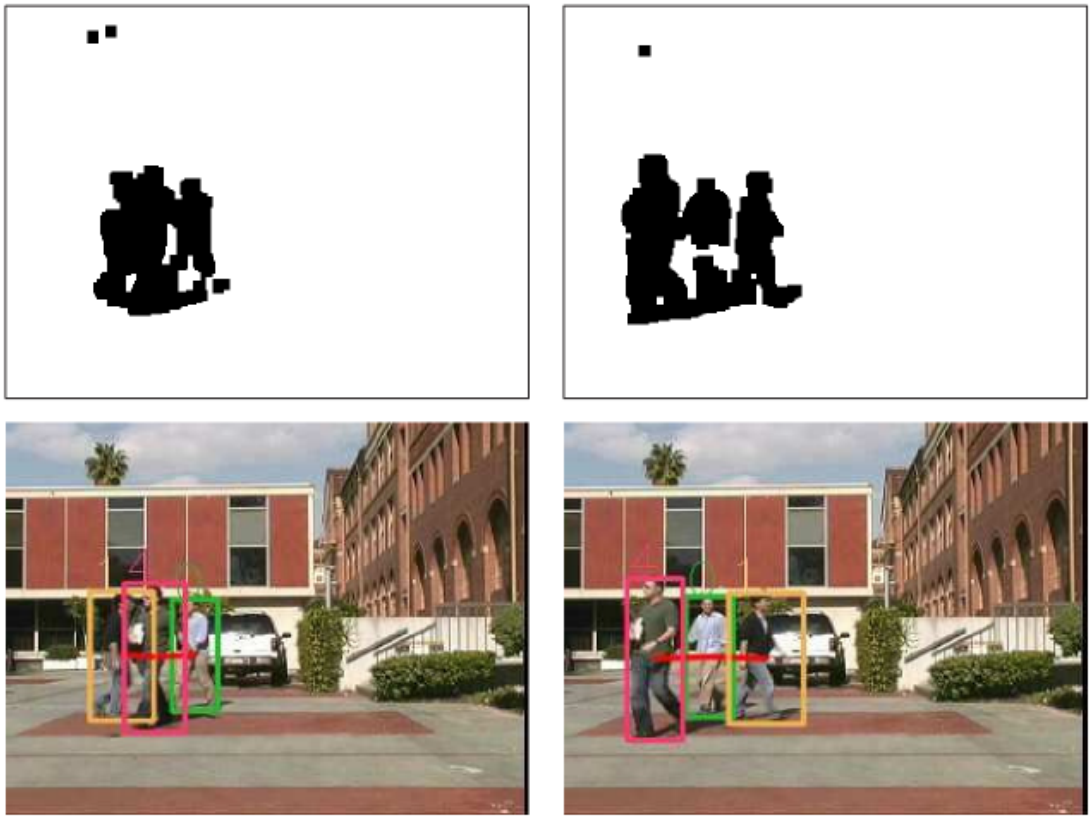

Abbildung 4.5: *Innerhalb der ‚Computer Vision' ist das Tracking von mehreren sich bewegenden Objekten interessant. Die Perzepte generieren sich dabei aus interessanten Regionen, so genannten Blobs. Neben der Verarbeitung sehr verrauschter Daten liegt eine Hauptaufgabe der Filter darin, Okklusionen zu erkennen. Aus [YMC07].*

In den letzten 15 Jahren fokussierte sich die Forschung aufgrund der stetig besseren und günstigeren Kamerasystemen auf das Tracking von mehreren Personen mittels visueller Sensorinformationen. Dabei werden Personen in potenziell interessanten Regionen (sogennante *Blobs*) erkannt [YMC07]. Innerhalb eines Zielobjekts können unter Umständen mehrere solcher Blobs ausgemacht werden. Mehrere Zielobjekte können aber auch zu einem großen Blob *zusammenfallen* (Überdeckungen). Die Abbildung 4.5 zeigt ein beispielhafte Bilderreihe solcher Untersu-

chungen. In den oberen Bildern sind Differenzbilder aus vergangenen Sensoreingaben zu erkennen. Dabei sind die Bereiche schwarz markiert, die eine Veränderung zum vorhergehenden Bild darstellen. Die unteren Bilder zeigen die dazugehörenden Bildinformationen und die entsprechenden Regionen von erhöhtem Interesse sind quadratisch umrandet. Die Farben deuten die entsprechende Unterscheidung der Objekte an.

Die bisherige Annahme, dass ein Objekt genau eine Messung hervorruft, ist bei solchen *regionalen* Wahrnehmungen nicht mehr verfolgbar [YMC07]. *Yu*, *Medioni* und *Cohen* gehen dieses Problem an, indem Sie neben den räumlichen (Regionen im Bild) ebenso zeitliche Informationen nutzen. In ihren Worten ausgedrückt, suchen sie das *räumlich-zeitliche globale Optimum* und bilden mit einem entsprechenden Zeitversatz Wissen über die Trajektorien von Zielobjekten [YMC07]. Andere aktuelle Forschungen versuchen das Erkennen von Objekten zu verbessern, um die Sensorinformationen bereits vorzuklassifizieren.

4.3 Zusammenfassung

Auch wenn in der RoboCup-Domäne bereits zuverlässige Möglichkeiten bestehen, den Roboter global auf dem Feld zu verorten [Göh09, Tas14], so können zukünftige Regeländerungen die Berechnung der globalen Position erschweren. Es ist beispielsweise denkbar, dass das Tor variable Pfostenabstände hat oder die Feldgrenzen variieren. Orientieren würde sich dies an einem Fußballspiel wie es Kinder auf dem Pausenhof austragen. Vier Rucksäcke reichen, um ein ganzes Spielfeld zu erstellen und ein Fußballspiel durchzuführen. Selbst wenn die Kinder eine Spielpause einlegen ist für Außenstehende das Spielfeld erkennbar.

Der Veröffentlichungen in der *Computer Vision* bieten einen umfangreichen Pool an Lösungsstategien. Auch hier wird konsequent von einer beschränkten Wahrnehmung ausgegangen, wie wir sie in unserem Umfeld vorfinden (vgl. 1.1). Vielversprechende Lösungen sind aber oftmals nicht tauglich für eine Echtzeit-Erfassung, da einige Ansätze nur durch A-Posteriori-Wissen zuverlässige Zustandsangaben treffen [YMC07]. Für die Videoüberwachung mag dies nicht entscheidend sein. In vielen Anwendungen für autonome humanoide Roboter ist der Echtzeit-Charakter ein besonderes Kritierium. Mit Hilfe der erkannten Objekten können beispielsweise nächste Schritte vorausgeplant werden. Es ist auch denkbar, dass Teamentscheidungen auf aktuelle Wahrnehmungen aufbauen [CKU+10, RLM+13].

Daher bieten vor allem die Arbeiten aus dem Bereich der autonomen Robotik vielversprechende Lösungsstrategien. Die Perzepte, die aus Laser-Sensoren erzeugt werden, beinhalten die ungefähre relative Position eines Objekts zum Roboter [SBFC03]. Aufgrund der wenigen Informationen die ein Laser-Sensor bietet, werden die erkannten Objekte als uniform angesehen. Die Pfosten im RoboCup-

Szenario sind ebenfalls nicht unterscheidbar und das gelieferte Perzept besteht lediglich aus der Position des Objekts relativ zum Roboter (vgl. 2.3.3). Daher sind vor allem die Arbeiten [SBFC03, IB96, KMA01] für die Entwicklung eines des Multi-Hypothesen-Tormodells interessant. Die Veröffentlichungen enthalten die Annahme, dass eine Messung stets ein Objekt repräsentiert. Die Kritik von [YMC07], dass 1-zu-1-Annahmen ein Schwachpunkt im Tracking darstellen, ist im RoboCup-Kontext unbegründet. Gelieferte Torpfostenperzepte sind entweder falsche Messungen oder repräsentieren genau einen Torpfosten. Diese Annahme bleiben keineswegs auf dem RoboCup beschränkt. Die Bestrebungen in der *Computer Vision*, individuelle Objekte zu erkennen zeigen, dass Lösungen mit dieser 1-zu-1-Annahme in Zukunft relevant sein werden.

5 Multi-Hypothesen-Tormodell

In diesem Kapitel wird die Implementation eines Tormodells vorgestellt, das im RoboCup Kontext verwendet werden soll. Angelehnt an die Anforderungen von *Reid* für ein Tracking mehrerer Objekte [Rei79], das im Kapitel 4 vorgestellt worden ist, soll der zugrunde liegende Algorithmus mit folgenden Schwierigkeiten umgehen können.

1. Umgang mit Falschmessungen (*false-positives*), d.h., Messergebnisse entsprechen in Wirklichkeit keinem Pfosten

2. *Neue* Pfosten sollen *getracked* werden

3. Pfosten im aktuellen Bildausschnitt können durch Überdeckung nicht erkannt werden

Abbildung 5.1: *Das line Bild zeigt die Situation während eines Spiels. Das Publikum sitzt sehr dicht am Spielfeldrand. Kurzzeitige Pfostensichtungen in Armen und Beinen sind möglich. Das rechte Bild zeigt aus der Roboterperspektive einen Torpfosten und gelbe Stühle im Hintergrund. Unter Umständen liefert die Bildverarbeitung ein dauerhaftes Torpfostenperzept in der Nähe der Stühle.*

Als Eingangsdaten des Algorithmus werden zweidimensionale Positionen der Torpfosten in relativen Roboterkoordinaten erwartet. Diese sogenannten Perzepte werden von dem vorhandenen visuellen Wahrnehmungssystem des NaoTH-Frameworks geliefert (vgl. 2.3.3).

Im Kapitel 6 wird gezeigt, dass diese Eingangsdaten Ungenauigkeiten unterworfen sind. Darüber hinaus zeigt die Erfahrung, dass der Roboter in der Realtität immer wieder Falschwahrnehmungen ausgesetzt ist. Diese Artefakte können insbesondere durch die Umgebung und wechselnde Lichtverhältnisse hervorgerufen werden. Zur besseren Vorstellung zeigt Abbildung 5.1 die Umgebung während eines Fußballspiels. Im linken Bild ist zu sehen, wie dicht das Publikum am Spielfeldrand sitzt. Je nach Lichtverhältnissen kann es zu irrtümlichen Wahrnehmungen von Pfosten in Armen und Beinen der Zuschauer kommen. Das rechte Bild zeigt eine Szene aus der Roboterperspektive. Gelbe Stühle im Hintergrund können unter Umständen dazu führen, dass die Bildverarbeitung permanent ein Pfostenperzept an dieser Stelle herausarbeitet.

Ein Modell das Torpfostenperzepte zusammengefasst, muss demnach robust gegenüber drei verschiedenen Arten von Unsicherheiten sein:

- Sensorrauschen aus der Bildverarbeitung und der Kinematik
- Spärliche Falschmessungen *(false-positives)*
- Systematischen Inkonsistenzen resultierend aus dichten Falschmessungen

Zum Umgang mit diesen Schwierigkeiten wird auf einen Multi-Hypothesen-Ansatz zurückgegriffen. In Kapitel 4 wurden dazu schon verwandte Arbeiten vorgestellt. Das Kapitel 6 und 7 wird zeigen, dass die Fehler der Sensoreingaben nicht normalverteilt sind. Das führt insbesondere dazu, dass die Verteilung des zu modellierenden Zustands nicht zwangsläufig normalverteilt sein muss. Zusätzlich ist der zu modellierende Zustand eines Pfostens nicht-linear. Ganz besonders deutlich wird dies beispielhaft im Abschnitt 7.6 anhand eines Zweikampfes zweier Roboter aufgezeigt werden. Ein geeignetes Filterungsverfahren für nicht-lineare nicht-normalverteilte Zustände ist der Partikelfilter [GSS93, IB96, PS99, SBFC03, VGP05]. Jeder einzelne Torpfosten wird mittels eines Partikelfilters repräsentiert und als Multi-Hypothese bezeichnet.

Der linke und der rechte Pfosten ist für den Roboter nicht am Aussehen zu unterscheiden. Darüber hinaus kann die Umwelt dichte Falschwahrnehmungen liefern. Um diese Mehrdeutigkeiten aufzulösen, wird jede einzelne Messung einer Multi-Hypothese zugeordnet, die diese Messung am besten vorhersagt. Dabei wird davon ausgegangen, dass eine einzelne Messung genau einer Multi-Hypothese entspricht. Sollten alle Vorhersagen unter einer gewissen Güte liegen, wird die Messung in einem Perzeptpuffer gespeichert, der zeitgesteuert automatisch geleert wird. Sollte sich in dem Puffer eine kohärente Menge von Messungen akkumulieren, die für die Bildung einer neuen Multi-Hypothese ausreichend sind, so wird eine neue Multi-Hypothese erzeugt und die Messungen aus dem Puffer entfernt. Dies soll insbesondere verhindern, dass spärliche Falschmessungen die einzelnen Multi-Hypothesen verrauschen.

Die einzelnen Schritte des Algorithmus werden im nachfolgenden Abschnitt detailliert dargestellt.

5.1 Mathematische Formulierung

Gegeben: Im Zeitschritt t_n erhält der Roboter eine Menge von Torpfostenperzepten $\{p_i\}_0^k$ mit

$$p_i = (\alpha_i, d_i) \in \mathcal{X}_p := [-\pi, \pi] \times \mathbb{R}_+.$$

Dabei ist d_i die Distanz und α_i der horizontale Winkel zum gesehenen Torpfosten im lokalen Polarkoordinatensystem des Roboters. In der Regel liegt die Anzahl der Perzepte k zwischen Null und Zwei. Darüber hinaus steht die Odometrie-Messung $o \in SE(2)$ zur Verfügung. Dabei ist

$$SE(2) := \{T : \mathbb{R}^2 \to \mathbb{R}^2 \mid T(x) = Ax + b, A \in SO(2), b \in \mathbb{R}^2\}$$

die spezielle Euklidische Gruppe (Gruppe aller Koordinatentransformationen, d.h., Rotation und Translation). Die Odometrie beschreibt die Roboterbewegung seit dem letzten Zeitschritt in 2D–Koordinaten, d.h. die relative Koordinatentransformation zwischen der letzten und der aktuellen Position des Roboters.

Gesucht: Das interne Modell eines einzelnen Pfostens wird durch eine Menge an Partikeln $H \subset \mathcal{X}_p$ repräsentiert. Der Zustand des Partikelfilters entspricht dabei der Multi-Hypothese. Der Partikelfilter ist nach oben (M) und unten (m) in der Anzahl seiner Partikel beschränkt: $m \leq |H| \leq M$. Das gesamte Modell besteht dabei aus einer Menge an Hypothesen $\mathcal{H} := \{H_1, ..., H_n\}$ und einem Perzeptpuffer $B \subset \mathcal{X}_p$. Aus diesen Multi-Hypothesen wird ein konkretes Tormodell extrahiert.

$$\text{Tormodell: } (p_L, p_R) = (\alpha_L, d_L, \alpha_R, d_R) \in [-\pi, \pi] \times \mathbb{R}_+ \times [-\pi, \pi] \times \mathbb{R}_+$$

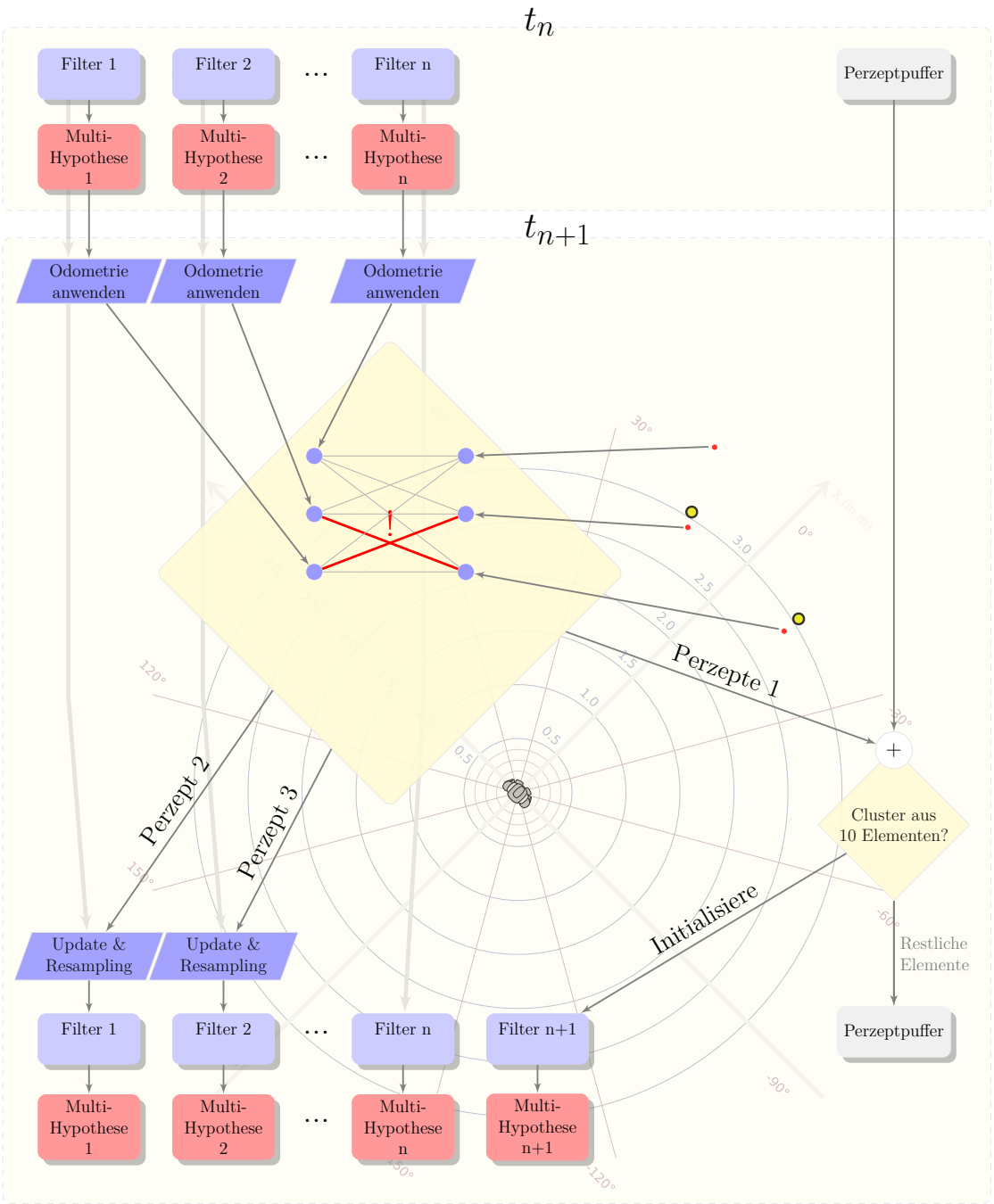

Abbildung 5.2: *Die Abbildung zeigt den Verlauf innerhalb des Zeitschrittes t_{n+1}. Die Vorhersage der Multi-Hypothesen für den nächsten Schritt wird getätigt (1). Die Perzepte werden mit den alten Multi-Hypothesen verglichen und gegebenenfalls auf die dazugehörigen Filter als Update angewendet (2). Andernfalls wird das Wissen in einem Puffer vorgehalten und bei einem genügend großen Cluster ein neuer Filter, also eine neue Hypothese, erzeugt.*

Algorithmus des Multi-Hypothesen-Tormodells

Durch ein neues Bild wird der Algorithmus ausgelöst. Neben den Perzepten von Torpfosten $P := \{p_1, ..., p_k\} \subset \mathcal{X}_p$ berechnet der Roboter Informationen über seine Odometrie $o \in SE(2)$.

1. *Vorhersage (Bewegungs–Update)* aller Partikelfilter aus $\mathcal{H}$ und des Perzeptpuffers B mittels Odometrieinformationen. Jedes Element der Menge wird mittels Odometrie o transformiert.

2. Update

   a) *Assoziation* Für alle eingehenden Perzepte $p \in P$ assoziiere, ob p einem existierenden Filter aus $\mathcal{H}$ zugeordnet werden kann. Existiert ein solches $H \in \mathcal{H}$, dann *update* H mittels der Sensordaten p und entferne p aus P ($P := P \setminus \{p\}$).

   b) Alle ungenutzten Perzepte werden dem Perzeptpuffer hinzugefügt $B := B \cup P$.

   c) Resampling der Partikel der vorhanden Filter $H \in \mathcal{H}$.

3. Manage

   a) Überprüfe für alle Filter $H \in \mathcal{H}$ ob ihre *Gewichtung* einen bestimmten Schwellwert übersteigt, ansonsten verwerfe den Filter ($\mathcal{H} := \mathcal{H} \setminus \{H\}$). Es werden dabei Gewichtungen für alle Filter aus den alten Durchschnittswerten berechnet.

   b) Entferne Beobachtungen, die eine festgeschriebene Zeit zurückliegen, aus dem Puffer B.

   c) Clustern des Puffers nach euklidischer Distanz und überprüfe, ob ein *großes* Cluster $C \subset B$ mit $|C| > m$ enthalten ist. Benutze ein solches Cluster um einen neuen Filter H zu initialisieren und füge es der Menge an Filtern hinzu $\mathcal{H} := \mathcal{H} \cup \{H\}$.

4. Extrahiere das Model basierend auf den Multi-Hypothesen $H \in \mathcal{H}$. Finde $H_1, H_2 \in \mathcal{H}$ so, dass $d := (\|E(H_1) - E(H_2)\|_2 - goalDistance)$ minimal ist. Falls die Abweichung d unter einer festen Schwelle liegt, dann sortiere die beiden Pfosten entsprechend der Annahme, dass der Roboter sich auf dem Feld Befindet und setze o.B.d.A. $p_L = E(H_1), p_R = E(H_2)$. Dabei ist $E(H_{1,2})$ der Erwartungswert der entsprechenden Multi-Hypothese, d.h. in unserem Fall der Mittelwert des Filters. Der Wert $goalDistance$ ist die bekannte Breite des Tores.

Algorithmus 3 : Multi-Hypothesen-Tormodell

Eingabe : P_n, o_n $\mathcal{H}_n$, B_n
Ausgabe : p_L^{n+1}, p_R^{n+1}, $\mathcal{H}_{n+1}$, B_{n+1}

```
1  B̂_n = predict(B_n, o_n)
2  foreach H ∈ H_n do
3  |  Ĥ_n = predict(H, o_n)
4  end
5  A = assoziiere(H_n, P)
6  foreach p ∈ P_n do
7  |  if A(p) ≠ ∅ then
8  |  |  update(A(p), p)
9  |  else
10 |  |  B = B ∪ {p}
11 |  end
12 end
13 C = largestCluster(B)
14 if |C| >= T_H then
15 |  H := H ∪ {C};
16 end
17 extractModel(H)
```

Im Schritt 2a kann jedes Perzept mit nur einem Pfosten assoziiert werden. Um das Modell des ganzen Tores zu schätzen, wählt man zwei Filter $H_1, H_2 \in \mathcal{H}$ mit der besten Gewichtung und korrekter Distanz zwischen den Pfosten eines Tores. Ein ganzes Tor kann somit nur abgeschätzt werden, wenn mindestens zwei Filter existieren und diese Bedingungen erfüllen. In diesem Fall wird angenommen, dass ein Pfosten dem Mittelwert eines Filters entspricht.

Die Grafik 5.2 zeigt den zeitlichen Ablauf der Schritte 1 und 2 des Algorithmus schematisch auf. Die Algorithmen 3 und 4 stellen das Multi-Hypothesen-Tormodell in Pseudocode dar.

5.2 Diskussion

In unterschiedlichen Ausprägungen haben es alle Multi-Hypothesen-Ansätze gemein, die nachfolgenden Kriterien genauer beleuchten zu müssen. Im Wesentlichen ist von besonderer Bedeutung, wann durch Sensoreingaben eine neue Hypothese initialisiert wird. Darüber hinaus ist die Technik zu diskutieren, wie Sensoreingaben mit vorhandenen Hypothesen assoziiert werden und ab wann man Hypothesen

Algorithmus 4 : assoziiere

Eingabe : Multi-Hypothesen $\mathcal{H}$, Perzepte P
Ausgabe : Menge der Assoziierungen A

```
1  if H = ∅ ∨ P = ∅ then
2  |  A := ∅
3  else
4  |  (H_0, p_0) := argmax_{(H,p)∈H×P} confidence(H, p)
5  |  if confidence(H_0, p_0) > T_c then
6  |  |  A := {(H_0, p_0)} ∪ assoziiere(H \ {H}, P \ {p})
7  |  else
8  |  |  A := ∅
9  |  end
10 end
```

verwirft. Zusätzlich muss man offenlegen, wie man aus einer Menge an Hypothesen eine geeignete Repräsentation eines Modells extrahiert.

Jede Technik für einen dieser Punkte bedingt gleichermaßen andere Punkte der Auflistung. Im Folgenden wird die Herangehensweise im Rahmen dieser Arbeit diskutiert und Referenzen auf mögliche Alternativen dargelegt.

5.2.1 Initialisierung

Das Pflegen vieler Filter kann in einer ressourcenbeschränkten autonomen Plattform aufgrund der eingeschränkten Rechenleistung zu Laufzeitproblemen führen. Daher ist es gerade im gewählten Arbeitsumfeld sehr wichtig, bei der Initialisierung auf Relevanz zu achten. Um dies zu gewährleisten puffern wir zunächst erhaltene Perzepte. Diese Technik hat sich gegenüber Falschmessungen und temporär falsch erkannten Objekten als sehr robust erwiesen.

Die Intialisierungsroutine ist ausbaufähig. Derzeit wird eine Hypothese $H \in \mathcal{H}$ bestehend aus $n \in \mathbb{N}$ Partikel initialisiert. Diese n Partikel entsprechen n Perzepten eines Clusters C aus dem Perzeptpuffer B: $C \subset B$ mit $|B| > n$.

Die Anzahl der Partikel eines Partikelfilters erhöht den Rechenaufwand, beeinflusst die Reaktivität des Filters und die Genauigkeit der Messung des zu filternden Zustandes. Die Erhöhung der Partikel die zur Initialisierung einer Hypothese aus dem Puffer benötigt werden, bedeuten eine deutlich trägere Initialisierungsphase. Zur unabhängigen Steuerung dieser Eigenschaft des Partikelfilters und des Initialisierungsschrittes, ist eine Entkoppelung wünschenswert.

5.2.2 Assoziation

Das Problem der Assoziation von erkannten Objekten und gepflegten Hypothesen ist weitreichend [IB96, YMC07, SBFC03]. Auch wenn die Implementation des SJPDA-Filters aus [SBFC03] praktizierbar erscheint, so ist ein kombinatorisch exponentielle Assoziation von allen Partikeln mit allen Messungen nur bedingt für eine ressourcenbeschränkte autonome Plattform umsetzbar (vgl. 4.2 und 2.2).

In unserem Umfeld werden aus Bilddaten einzelne Objekte erzeugt. Aufgrund dieses Charakters kann man begründet davon ausgehen, dass ein Perzept genau einer Hypothese zuordenbar ist [YMC07].

5.2.3 Löschen und Zusammenführen

Die Löschung beruht auf dem Umstand, dass Hypothesen, die für eine bestimmte Zeit keine Bestätigung durch eingehende Perzepte erfahren, an Gewichtung verlieren und daher gelöscht werden. Es ist somit nicht nötig, dass Hypothesen in diesem Ansatz zusammengeführt werden. Es ist davon auszugehen, dass Hypothesen, die potenziell das gleiche Objekt repräsentieren, ohnehin auch die selben Perzepte zugeordnet bekommen. Aufgrund der Annahme, dass ein Perzept genau einer Hypothese zugeordnet wird, hat die praktische Umsetzung gezeigt, dass eine der Hypothesen mit der Zeit an Gewichtung verliert und gelöscht wird.

5.2.4 Modellextraktion

Aus mindestens zwei vorhandenen Hypothesen wird das Modell extrahiert, das am wahrscheinlichsten einem Tor entspricht. Man spricht dabei von der *Maximum-Likelihood-Methode*.

Diese Heuristik hat sich in der praktischen Anwendung bewährt.

6 Empirische Analyse der visuellen Wahrnehmung

Wie im Abschnitt 1.2 beschrieben, betrachten wir das visuelle Wahrnehmungssystem des Roboters als eine Art *Blackbox*, bestehend aus der Bildverarbeitung und dem kinematischen Modell des Roboterkörpers. Die Bildverarbeitung liefert die Koordinaten und Ausdehnung eines Objektes im Bild in Pixelkoordinaten. Das kinematische Modell des Roboters wird anhand der gemessenen Gelenkpositionen und Inertialsensoren (Gyrometer und Beschleunigungssensor) berechnet. Aus diesem Modell lässt sich die relative Position und Ausrichtung der Kamera herleiten. All diese Bestandteile werden als ein geschlossenes System der visuellen räumlichen Wahrnehmung betrachtet.

Unter der Annahmen, dass sich das im Bild gefundene Objekt auf dem ebenen Boden des Spielfelds befindet, lässt sich eine relative Position des Objekts schätzen. Diese Position wird als Perzept bezeichnet. Perzepte sind räumliche Wahrnehmungen der Objekte aus der Umgebung des Roboters. Im Abschnitt 2.3.3 ist genauer beschrieben, wie der Roboter Perzepte erhält. Darüber hinaus enthält [Mel10, Kapitel 2] detaillierte Ausführung zur Distanzbestimmung mit Referenzobjekten.

Objekte die aus einem Bild extrahiert werden, sind mit bestimmten Fehlern behaftet. Offensichtlich gehen in eine solche Messung zweierlei Fehler ein. Zum einem die Fehler in der Bildverarbeitung und Objekterkennung selbst, und zum anderen die Projektionsfehler, die aus der Schätzung der kinematischen Kette resultieren. Dieser Abschnitt widmet sich einer experimentellen Untersuchung dieser Fehler. Ohne Beschränkung der Allgemeinheit wurde darüber hinaus nur die obere Kamera des Roboters berücksichtigt (vgl. 2.2).

6.1 Experiment I: Objekterkennung in Abhängigkeit der Objektposition im Bild

In diesem Experiment wird die Abhängigkeit der Objekterkennung von der Position des Objektes im Bild untersucht. Es wird dabei sowohl der Winkel, als auch die Entfernung vom Objekt mit vier unterschiedlichen Robotern untersucht.

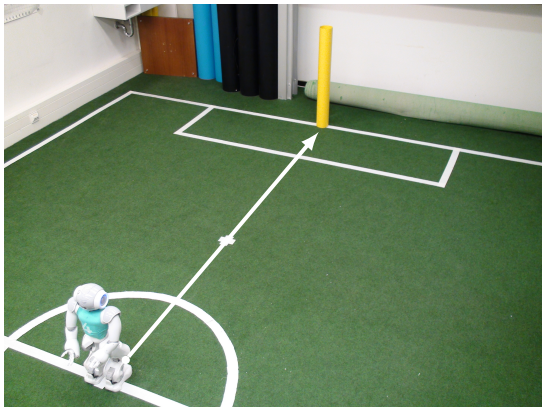

Abbildung 6.1: *Der Roboter steht während des Experiments an einem festen Punkt mit einem festen Abstand von* 3 m *zum Pfosten. Der weiße Pfeil symbolisiert die untersuchte Entfernungsmessung. Gemessen wird der Abstand zum Ball, ausgehend von der Projektion des Körperschwerpunktes auf dem Boden.*

Versuchsaufbau

Der Roboter steht während des Experiments auf einem festen Punkt 3 m vom Torpfosten entfernt. Der Pfosten ist ebenfalls stationär, lediglich der Roboterkopf bewegt sich im Versuchsverlauf. Das Bild der Abbildung 6.1 zeigt den Versuchsaufbau.

Mit einer äußerst langsamen Kopfbewegung bewegt sich der Kopf von der eine zur anderen Seite. Beim Erreichen der festgelegten maximalen horizontalen Ausrichtung des Kopfes, bewegt sich der Kopf minimal von oben nach unten und bewegt sich erneut horizontal bis zur gegenüberliegenden maximalen Ausrichtung. Der Roboter *scannt* also den sichtbaren Bereich ab. In Abbildung 6.2 ist der Experimentverlauf anhand der Egoperspektive des Roboters dargestellt. Die schwarzen Pfeile zeigen den Verlauf der Position des Torpfostens im Bild. Der blaue Rahmen entspricht der Kamera des Roboters. Die Extrembereiche der Kopfauslenkung sind so gewählt, dass der Torpfosten im gesamten Bildbereich erscheinen könnte. Die Abbildung 6.2 lässt dies erkennen, da die potenzielle Position des Torpfostens (schwarze Pfeile) den blauen Kamerabereich überlagern.

Der Roboter speichert während dieses Durchlaufs den resultierenden Abstand und Winkel zum Torpfosten kontinuierlich. Die verwendete Kopfgeschwindigkeit ist dabei so gering, dass das Experiment einer Messung im Stillstand gleichkommt.

Resultate

Abbildung 6.3 zeigt das Resultat des Experiments mit vier unterschiedlichen Robotern. Jede Zeile besteht dabei aus einer Messung. Die linke Graphik visualisiert die Distanz zum Torpfosten, die rechte Graphik den Winkel. Jeder Punkt entspricht dabei einer Messung. Die Farbe des Punktes beschreibt den Abstand bzw. den Winkel zum Torpfosten zum Roboter. Die rechte Skala gibt Aufschluss über die

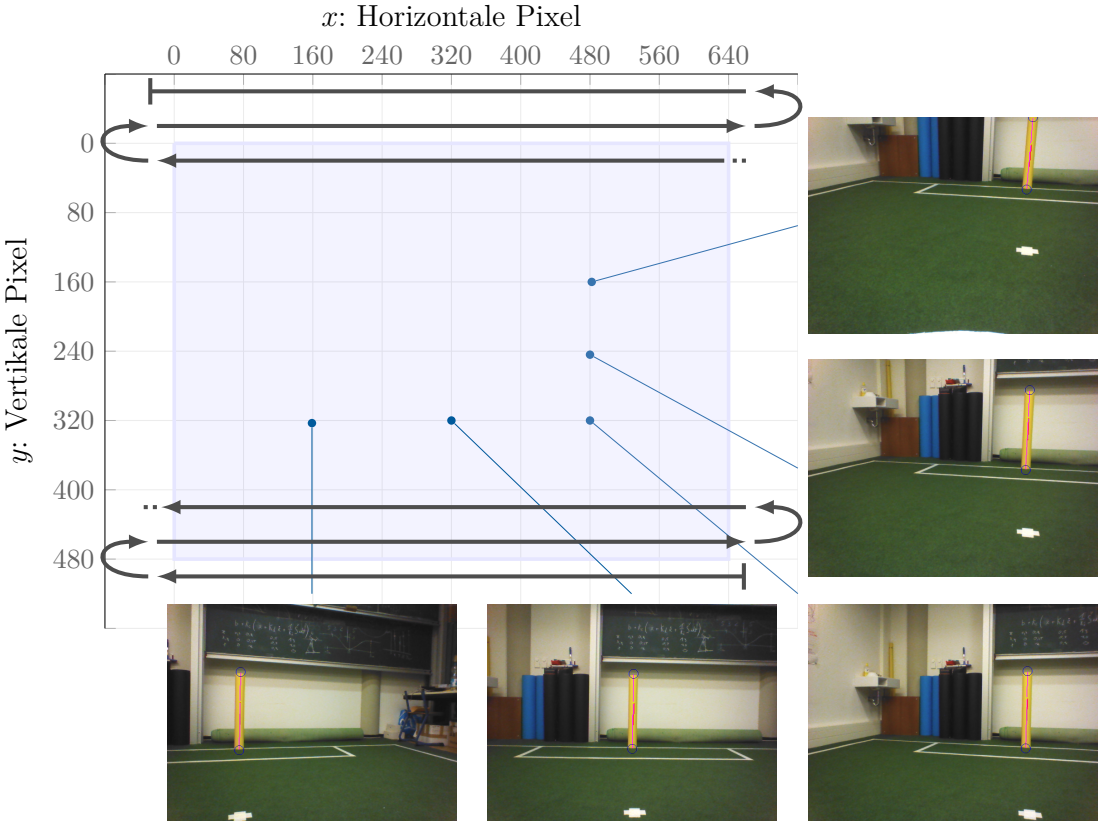

Abbildung 6.2: *Der schwarze Pfad in der Graphik symbolisiert die Torpfostenposition im Bild. Der blaue Bereich stellt die Gesamtheit der Pixel des Kamerabereichs dar. Die Position und der Winkel zum Torpfosten wird bei jeder Erkennung gespeichert. Die Geschwindigkeit ist dabei äußerst langsam.*

exakte Entfernungsmessung in Metern. Verschiedene Beobachtungen sind diesem Experiment zu entnehmen.

1. Das Objekt wurde in bestimmten Bereichen gar nicht erkannt, auffällig sind vor allem die Randbereiche.

2. Es zeichnen sich im mittleren Bereich des Bildes zusammenhängende Linien ab. In anderen Bereichen scheint der Objektmittelpunkt leicht zu variieren, trotz konstanter Kopfbewegung.

3. Die genannten Linien sind nicht horizontal.

4. Die Abstandsmessung folgt dem gleichen Verlauf wie in Punkt 3. Wenn die Linien von rechts oben nach links unten Verlaufen, nimmt die Abstandsmessung beispielsweise zu.

5. Benachbarte Punkte sind ähnlich gefärbt, Abstandsmessungen sind demnach lokal ähnlich.

6. Entlang einer gedachten senkrechten Linie nimmt der Abstand zu.

Die Form des erkennbaren Bereichs des Objekts lässt vermuten, dass die Beobachtung 1 einem sphärischen Abbildungsfehler der Kamera zuzuordnen ist. Anders ausgedrückt, ist die Beleuchtung des Bildes nicht einheitlich genug, um das Objekt in allen Bereichen gut zu erkennen. Eine Ausnahme bildet hier der zweite Versuch.

In Abschnitt 2.3.3 wurde bereits ein sphärischer Fehler angesprochen und aufgezeigt, dass die Bildausleuchtung nicht über den gesamten Bildbereich konstant ist.

Die Beobachtungen 2 lässt sich auf die Bildverarbeitung (Abschnitt 2.3.3) zurückführen. Es ist zu erkennen, dass in den Randbereichen der Kamera der Mittelpunkt des Objekts nicht ähnlich präzise auf einen Pixel zuzuordnen ist, wie es im Zentrum der Fall ist. Der Einfluss dessen auf den eigentlichen Messfehler ist aber äußerst gering.

Die weiteren Beobachtungen (Punkt 3–6) sind indes aufgrund ihrer Struktur geometrischer Natur. Dies lässt sich darin erkennen, dass ihr Verlauf nur über das gesamte Bild erkennbar ist, nicht aber in lokalen Umgebungen. In den beiden oberen linken Graphiken der Abbildung 6.3 ist von einer positiven Rotation der Kamera auszugehen. Dies erklärt den Entfernungsanstieg von rechts oben nach links unten entlang einer horizontalen Linie. Im vorletzten Vesuch ist die Rotation der Kamera negativ, im letzten Versuch entsprechend neutral.

Auch die Tendenz, dass der Abstand von oben nach unten zunimmt (Punkt 6), kann durch einen geometrischen Fehler beschrieben werden. In [Mel10, Kapitel 2] sind dazu die nötigen geometrischen Überlegungen notiert.

Die jeweils rechte Graphik eines jeden Versuchs visualisiert den entsprechenden Winkel zum Torpfosten. Auch bezüglich der Winkelmessungen sind Abhängigkeiten zur Rotation der Kamera auszumachen. Des Weiteren sind auch die Winkelmessungen lokal ähnlich und untere Randbereiche sind ebenfalls größeren Fehlersprüngen unterworfen.

Im Allgemeinen bleibt festzuhalten, dass die Winkelmessung sehr akkurat ist. Der größte Fehler ist mit ca. 3,4° im zweiten Versuch auszumachen. Die Distanzunterschiede sind mit über 1,5 m bereits beträchtlich. Bei einem tatsächlichen Abstand von ungefähr konstant 3 m zum Objekt, entsprechen die Schwankungen der Distanzmessung bereits der Hälfte der Entfernung.

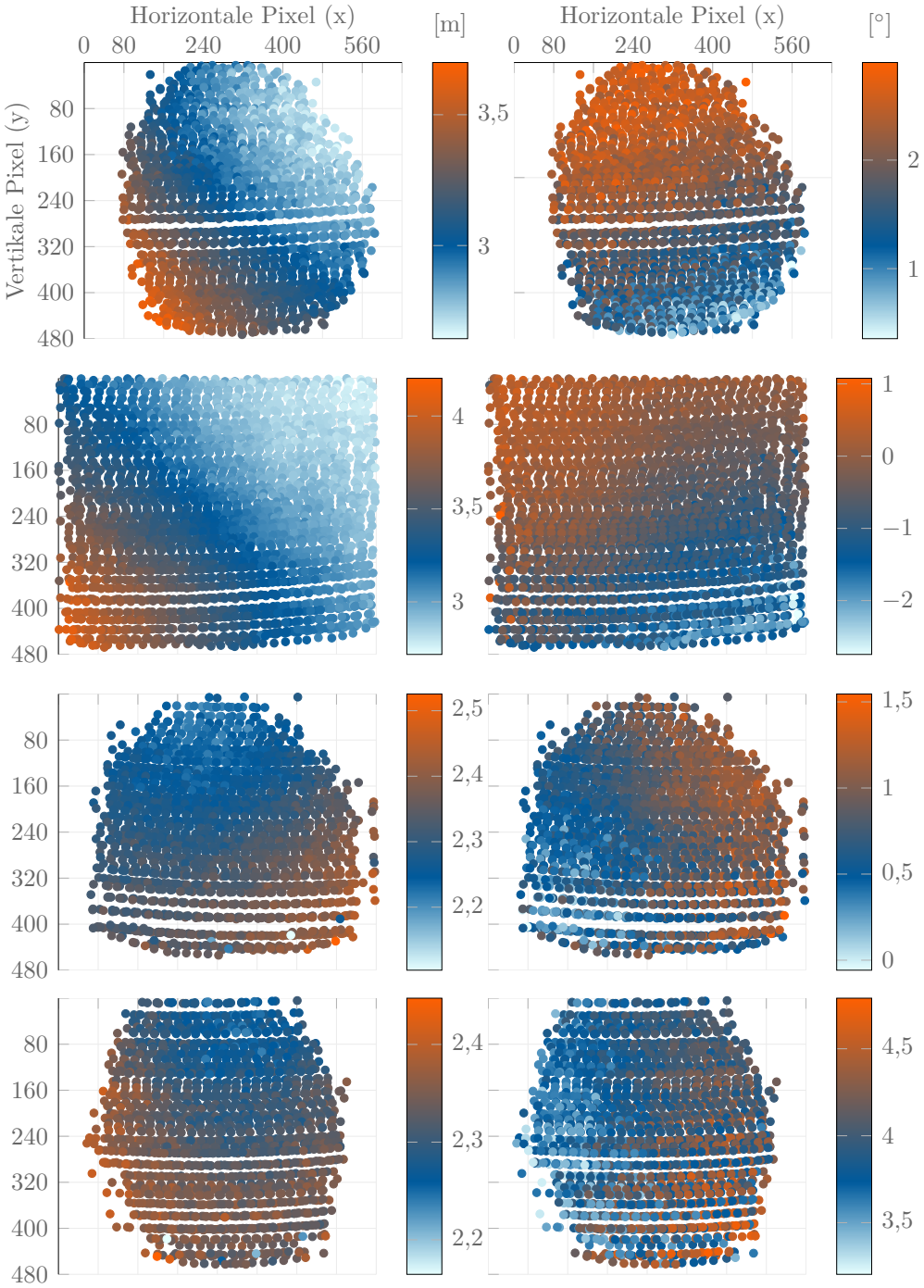

Abbildung 6.3: *Jeder Punkt entspricht einer Torpfostenerkennung. Jede Zeile visualisiert die Messungen für einen Roboter. Die jeweils linke Graphik zeigt die Distanz zum Torpfosten, die rechte Graphik den Winkel. Die Farbe eines Punktes symbolisiert die Distanz bzw. den Winkel. Die Veränderung beider Größen erscheint fließend. Lediglich im unteren Teil des Bildes kommt es zu hohen Schwankungen. Die Unterschiede in der Winkelmessung fallen gegenüber der Distanzmessung deutlich geringer aus.*

6.2 Experiment II: Objekterkennung in Abhängigkeit der Kopfposition

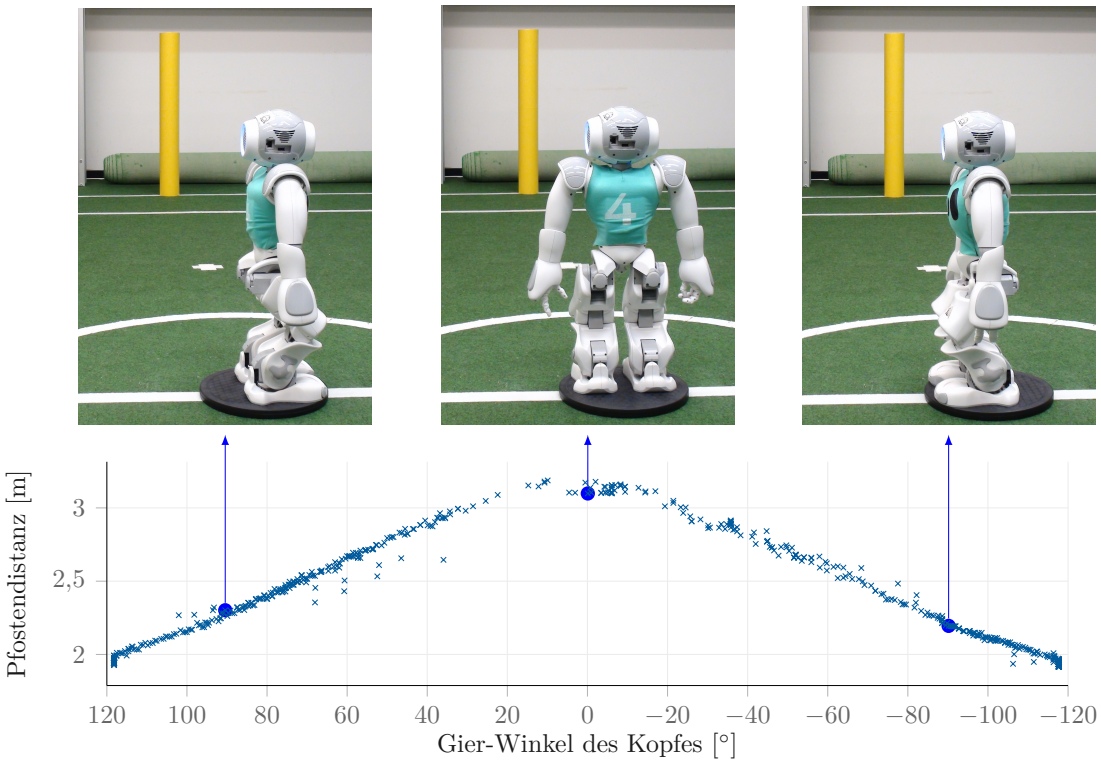

Abbildung 6.4: *Der Körperschwerpunkt des Roboters ist zentriert auf einer drehbaren Scheibe ausgerichtet. Ein Torpfosten ist* 3 m *entfernt. Der Fußpunkt des Torpfosten wird vom Roboter im Bild zentriert. Der Roboter schätzt die Entfernung des Torpfostens bezüglich seiner Projektion des Körperschwerpunktes auf den Boden. Der Graph zeigt die ermittelte Distanz in Abhängigkeit vom Gier-Winkel des Roboterkopfs. Die oberen Abbildungen zeigen die Position des Roboters bezüglich bestimmter Messungen.*

Dieses Experiment soll das Verständnis über die genutzte Sensorik des Roboters erweitern. Zusätzlich zum ersten Experiment aus Abschnitt 6.1 wird untersucht, inwieweit Projektionsungenauigkeiten aufgrund unterschiedlicher Ausrichtungen des Körpers zu erwarten sind.

Versuchsaufbau

In diesem Experiment ist die Position des Torpfostens und die des Roboters fixiert. Der horizontale Abstand vom Roboter zum Torpfosten beträgt erneut 3 m.

Der Roboterschwerpunkt ist zentriert auf einer drehbaren Scheibe ausgerichtet. Der Roboter zentriert während des Versuchs den Torpfosten im Kamerabild. Die Scheibe wird so gedreht, dass der Roboter jede mögliche horizontale Kopfausrichtung (jeden Gier-Winkel des Kopfes) einnimmt. Die oberen Bilder der Abbildung 6.5 zeigen den Versuchsaufbau.

Resultate

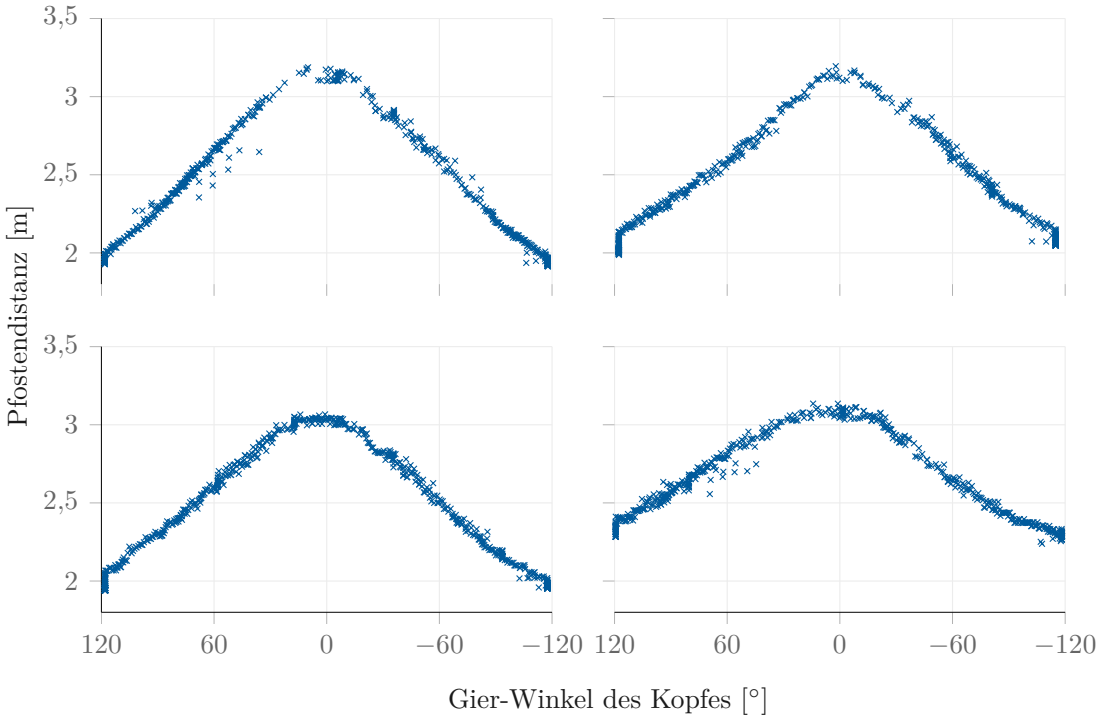

Abbildung 6.5: *Der Körperschwerpunkt des Roboters ist zentriert auf einer drehbaren Scheibe ausgerichtet. Ein Torpfosten ist 3 m entfernt. Der Fußpunkt des Torpfosten wird vom Roboter im Bild zentriert. Der Roboter schätzt die Entfernung des Torpfostens bezüglich seiner Projektion des Körperschwerpunktes auf den Boden. Der Graph zeigt die ermittelte Distanz in Abhängigkeit vom Gier-Winkel des Roboterkopfs. Die oberen Abbildungen zeigen die Position des Roboters bezüglich bestimmter Messungen.*

Das Experiment wurde an vier unterschiedlichen Robotern durchgeführt. Die Ergebnisse sind in den Graphen der Abbildung 6.5 dargestellt und offenbaren einen offensichtlich systematischen Fehler.

Für alle Roboter ist die Entfernung des Torpfostens umso geringer, je weiter der Roboterkopf betragsmäßig ausgelenkt wird. Da der tatsächliche Abstand zum Torpfosten während dieses Versuchs konstant geblieben ist und sich die Tendenz

bei allen Robotern abzeichnet, kann von einem systematischen Fehler in der Berechnung der Distanz ausgegangen werden.

6.3 Zusammenfassung

Die Experimente offenbaren bereits trotz ihres abgeschlossenen Charakters deutlich unterschiedliche Messungsungenauigkeiten für verschiedene Kameras und Roboter. Für Roboter, die sich in Bewegung befinden, ist mit weiteren Messungseinflüssen zu rechnen. Die Ballsuche während eines Spiels beispielsweise wird mit einer vielfachen Kopfgeschwindigkeit vorgenommen. Während des Laufens ist mit zusätzlicher Beeinflussung durch die instabile Lage des Roboters zu rechnen.

Abbildung 6.6: *Diese Abbildungen zeigen die autonome Messungsphase des Roboters zum Kalibrieren der Kamera in Abhängigkeit der kinematischen Konfiguration [KRL14]. Die oberen Bilder zeigen verschiedene Konfigurationen der Gelenke. Ebenfalls gut zu erkennen ist das fest justierte Schachbrettmuster am Fuß des Roboters. Die unteren Bilder zeigen die gesehenen und projizierten Punkte, die als Grundlage zur Kalibrierung dienen.*

Ein adäquates allumfassendes Sensormodell kann nur herausgearbeitet werden, wenn dieses mindestens die Ungenauigkeiten aller Kameras und Roboter mit einfließen lässt.

Zur Lösung dieser Problematik ließe sich die Überlegung heranziehen, die Lage der Kameras der Roboter bezüglich ihrer kinematischen Kette zu kalibrieren. So

wären die Perzepte aller Roboter vergleichbar und Modellierungsaufgaben würden sich stark vereinfachen.

Es existieren bereits zahlreiche Ansätze bezüglich solcher Kalibrierungsmethoden. Eine Übersicht bietet die Veröffentlichung *An overview of robot calibration* von Elatta, Gen, Zhi, Daoyuan und Fei [EGZ+04]. Umsetzungen dieser Ansätze lassen sich auch im Kontext des RoboCups finden. Einige Teams setzen auf eine halb automatische Lösung. Hierzu werden willkürliche Punkte auf den Feldlinien markiert. Diese Daten werden genutzt, um mit einem Gauß-Newton-Optimierer die extrinsischen Kameraparameter zu schätzen [RLM+13]. Weiterführungen dieses Ansatzes lassen den Roboter die Punkte auf dem Feld selbst erkennen und die entsprechenden Kameraparameter schätzen [RLM+14].

Ein weiterer aktueller Ansatz besteht darin, den Roboter mit zwei Platten mit Schachbrettmustern auszustatten, und ihn völlig autonom die Kalibierung vornehmen zu lassen. Der Roboter beobachtet die Schachbretter und ändert dabei seine kinematische Konfiguration. Abbildung 6.6 ist der Veröffentlichung entnommen und zeigt exemplarische Bilder aus einem Durchlauf der Messungsphase. Verschiedene Gelenkansteuerungen werden angenommen und das tatsächliche Schachbrett mit der Projektion durch aktuelle Kameraparameter verglichen. Angeschlossen wird eine Optimierungsphase, die Kalibrierung wird dabei als ein nicht lineare Ausgleichungsrechnung beschrieben [KRL14].

Ein weitere Möglichkeit die Messungenauigkeiten zu minimieren ist es, weiteres Wissen über die Umwelt einfließen zu lassen. Im Kapitel *Visuelle Distanzbestimmung anhand von Referenzobjekten* der Diplomarbeit von Heinrich Mellmann wird darauf genauer Bezug genommen [Mel10]. Durch das Ausnutzen von geometrischem Wissen am Beispiel der Ballgröße kann die Messgenauigkeit verbessert werden.

7 Empirische Analyse zur Bewegungswahrnehmung

Im Abschnitt 2.3.2 wurde das kinematische Modell des Roboters diskutiert. Insbesondere wurde die Berechnung des zurückgelegten Wegs (Odometrie) des Roboters im NaoTH-Framework vorgestellt.

In den folgenden Experimenten untersuchen wir die Güte der vom Roboter berechneten Odometrie. Dazu wird die Odometrie mit dem tatsächlich zurückgelegten Weg verglichen, der mit Hilfe des Trackingssystems (vgl. 2.4) erfasst wird. Der Roboter ist in der Lage sich omnidirektional zu bewegen. In den ersten beiden Experimenten wird untersucht, wie sich die Fehler einzelner Komponenten des Laufens verhalten. Dazu wird das Geradeauslaufen und das Drehen um die eigene Hochachse anhand von vier Robotern analysiert. Die Kombination dieser Laufrichtungen und die Auswirkungen auf die Genauigkeit der Odometrie werden in den letzten Experimenten examplarisch an einem Roboter aufgezeigt.

In den weiteren Experimenten 7.3, 7.5 und 7.6 werden exemplarisch einige spielnahe Situationen betrachtet.

7.1 Experiment I: Konstante Geradeausbewegung

In diesem Experiment wird untersucht, wie akkurat der Roboter seinen Weg berechnen kann, wenn er eine minimale lineare gleichmäßige Bewegung ausführt. Im tatsächlichen Einsatz ist der Roboter in der Lage, seinen Lauf sich ändernden Umwelteinflüssen anzupassen. In diesem isolierten Experiment soll der Roboter nur eine bestimmte Trajektorie ablaufen.

Der Roboter bekommt den Befehl geradeaus zu laufen. Das Experiment geht dabei etwa 25 Sekunden. Die Abbildung 7.1 zeigt den Verlauf der Position. Die tatsächliche Position ist rot, die abgeschätzte Position blau markiert. Der Startpunkt befindet sich beim Punkt $(0, 0)$.

Die Abbildung 7.2 zeigt die Messungsabweichungen im Verlauf des Experiments. Die obere Graphik der Abbildung 7.2 zeigt die Abweichung der x-Komponente im Verlauf der Zeit. Es ist zu erkennen, dass die Roboter ihren zurückgelegten Weg unterschätzen. Hierbei ist davon auszugehen, dass der Schwerpunkt nur unzureichend

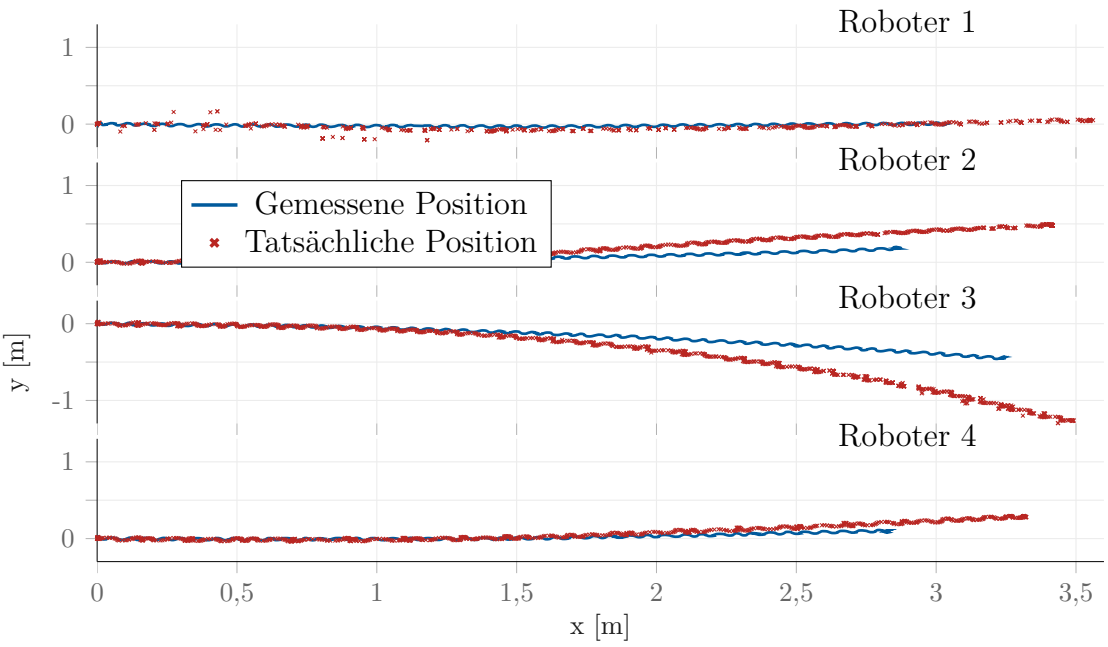

Abbildung 7.1: *Vier Roboter bekommen den Befehl, geradeaus zu laufen. Die Abbildung zeigt dabei ihre tatsächlichen zurückgelegten Wege (rot) und die berechnete Odometrie (blau). Anfänglich scheint die Odometrie bei allen sehr akurat zu sein. Es ist aber zu erkennen, dass alle Roboter sich zu langsam* einschätzen. *Es scheint die Odometrie der Ground Truth nachzuhängen.*

berechnet wird. Eine Überprüfung der Abmessungen der Roboter zur Berechnung des kinematischen Modells wäre eine erste Untersuchung.

Die untere Abbildung zeigt die Abweichung der y-Komponente. Es sind keine Regelmäßigkeiten zu erkennen.

Es ist zu erkennen, dass sich beide Fehler über die Zeit akkumulieren.

Drei von vier Roboter waren in der Lage, ihre Position nach 24 Sekunden auf 20 cm genau zu bestimmen. Das entspricht etwa ihrem eigenen Körperumfang und ist eine sehr hohe Genauigkeit. Ein Tendenz nach rechts auszuweichen ist zudem zu erkennen. Roboter 1 war bewegte sich anfänglich nach links, um dann während des Verlaufs nach rechts zu korrigieren. Ein rutschen ist hier an dieser Stelle denkbar, konnte aber nicht beobachtet werden. Wenn man das Laufverhalten exakt analysieren möchte, so muss dieser Durchlauf mit mehreren Robotern mehrere Male wiederholt werden. Dazu ist darauf zu achten, dass die Roboter stets im abgekühlten Zustand beginnen, um eine Vergleichbarkeit herzustellen. Eine Veränderung des Laufverhaltens über die Zeit und mit steigender Temperatur der Gelenke ist auch aussagekräftig, um die Modellierung über ein gesamtes Spiel zu verbessern.

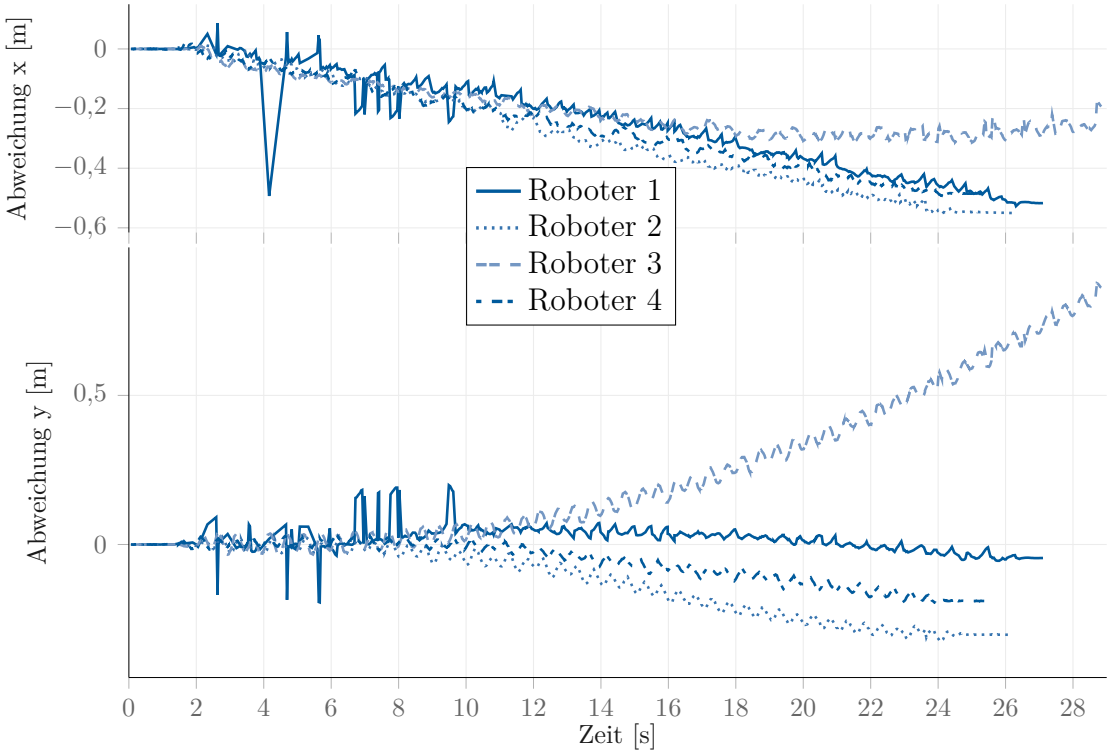

Abbildung 7.2: *Abweichungen zwischen den Trajektorien bezüglich der Odometrie und der Ground Truth. Dargestellt sind die x- und y-Komponenten.*

Für diese Arbeit lässt sich erkennen, das selbst bei einer kleinen Stichprobenmenge unvorhergesehene Ereignisse eintreten können. Ein offensichtlicher nichtlineare Prozess ist zudem zu erkennen.

7.2 Experiment II: konstante Drehung um die eigene Achse

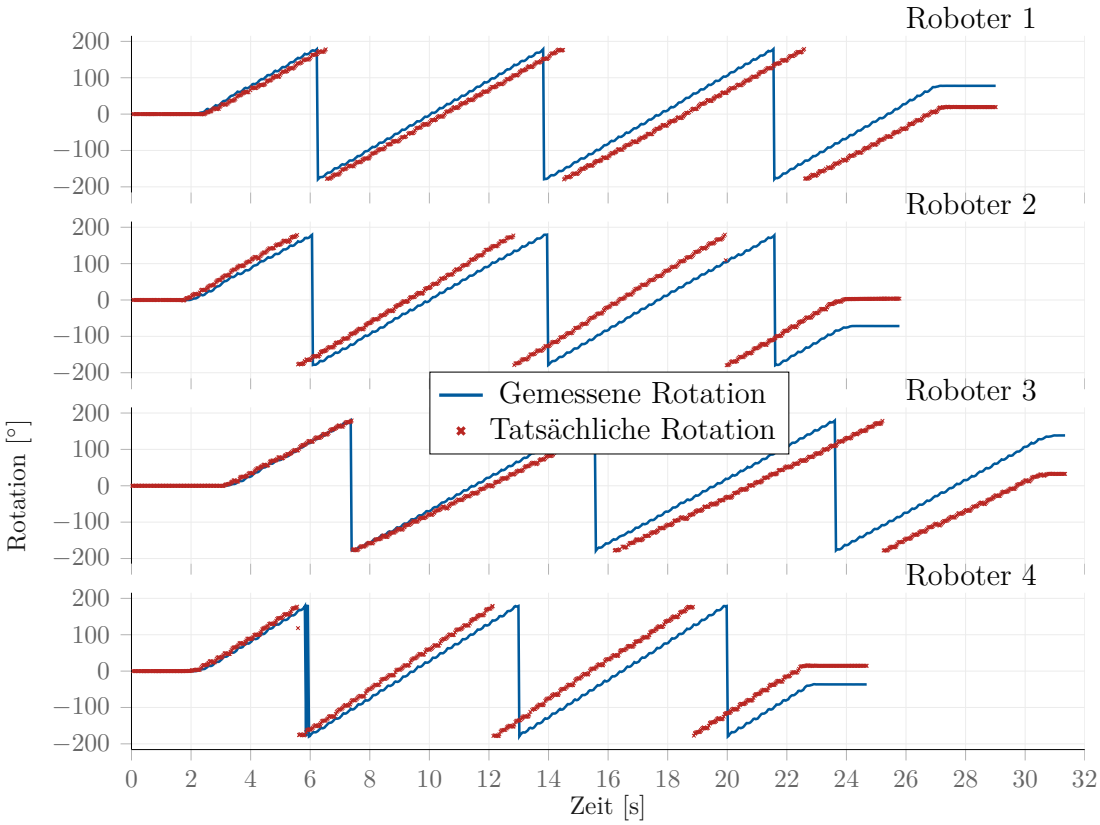

Abbildung 7.3: *Vier Roboter drehen sich etwa drei Mal um die eigene Achse. Dabei wird die Rotation von den Robotern berechnet. Seine Position verändert der Roboter dabei nur marginal. In dieser Stichprobe berechnen zwei Roboter ihre Drehung zu schnell, zwei andere zu langsam.*

Auch in diesem Experiment soll der Roboter eine isolierte gleichmäßige Bewegung ausführen. Der Roboter dreht sich um die eigene Achse, der Fokus ist auf die Veränderung seiner Rotation gerichtet.

In Abbildung 7.3 ist zu erkennen, dass jeweils zwei Roboter ihre Rotation überschätzen bzw. unterschätzen. Die Abweichung ist in Abbildung 7.4 abzulesen. Der Betrag der Abweichung ist dabei für alle Roboter mit 50°ähnlich.

Interessant ist der Vergleich das Verhalten der selben Roboter beim Ausführen einer Rechtsdrehung. Hierbei ist zu erwähnen, dass das Experiment zur Rechtsdrehung etwa 4 Minuten nach der Linksdrehung ausgeführt wurde.

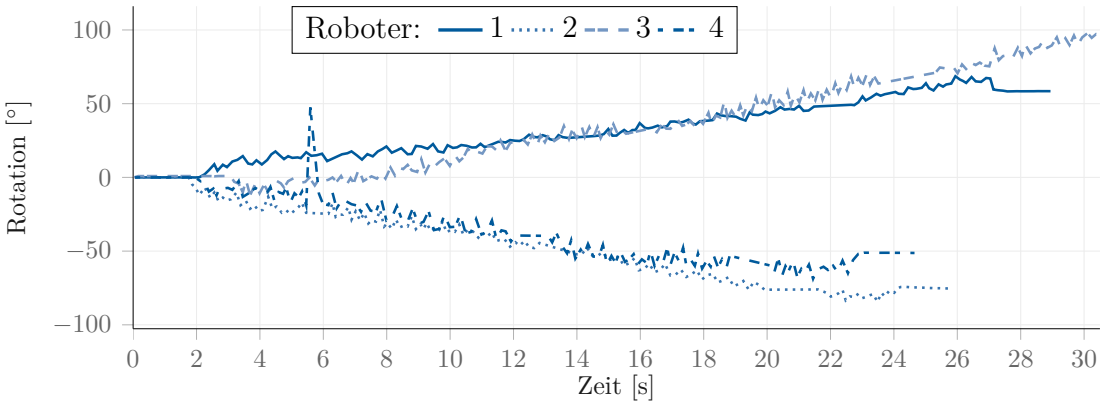

Abbildung 7.4: *Die Abweichungen der Rotation wird über die Zeit aufgezeigt. Jeweils zwei Roboter haben ihre Rotation über- bzw- unterschätzt. Die absolute Abweichung ist etwa gleichgroß. Die Roboter sind am Ende des Experiments um über 50° verschätzt.*

In der Abbildung 7.5 ist der Durchgang für die selben vier Roboter in der gleichen Roboter dargestellt. Auf dem ersten Blick ist zu erkennen, dass die Abweichung deutlich geringer ausfällt als bei einer Linksdrehung.

In Abbildung 7.6 ist die Abweichung der tatsächlichen Rotation und der berechneten Rotation dargestellt. In dieser Graphik ist zu erkennen, dass sich die Tendenzen unterscheiden. Ein Roboter schätzte seine Rotation zu langsam ein.

Im Allgemein lässt sich erkennen, dass eine Rechtsdrehung zuverlässiger als eine Rechtsdrehung. berechnet werden kann. Nach etwas 25 Sekunden verrechnet sich der Roboter bei einer Linksdrehung um mehr als eine viertel Drehung. In Anbetracht der Tatsache, dass ein Roboter durchaus eine halbe Minute kein Tor sehen kann, weil er auf dem Ball konzentriert ist, ist dies eine wichtige Information.

Die genutzte Version des Naos verfügt zudem nicht um Gyrometer mit z-Achse. Das NaoTH-Framework muss der berechneten Rotation vertrauen.

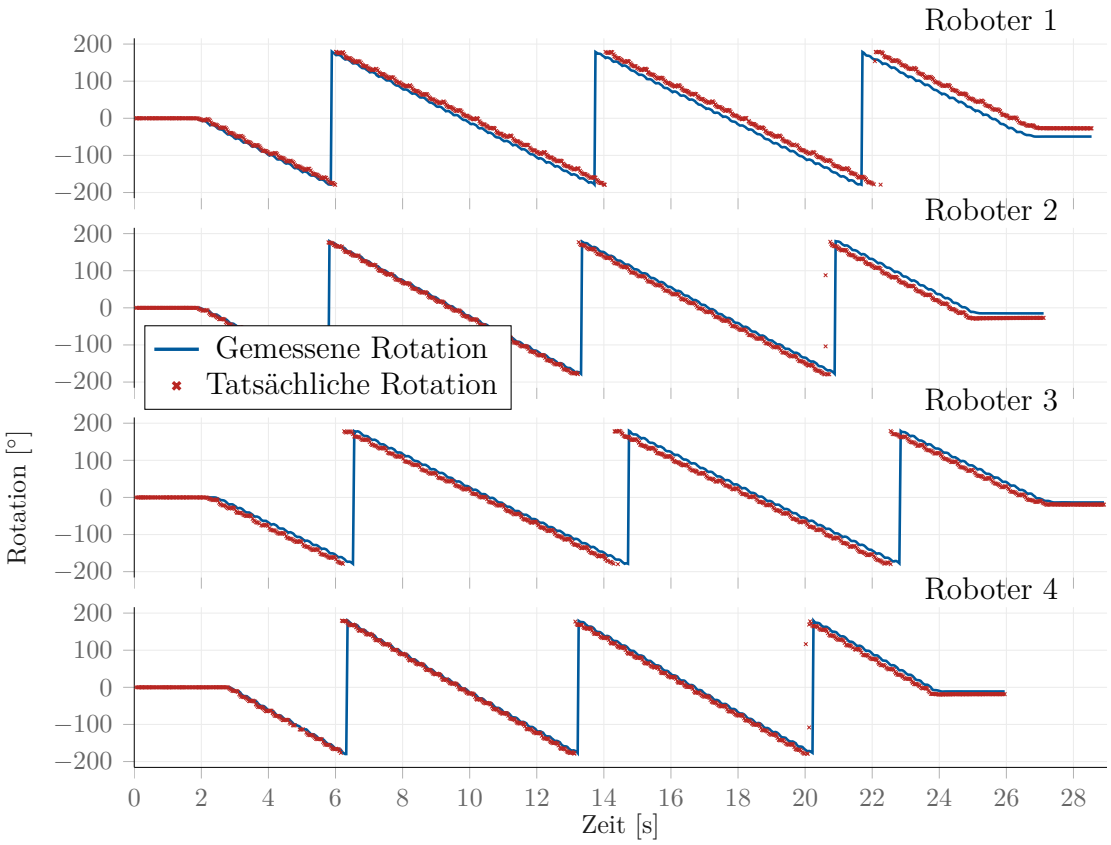

Abbildung 7.5: *Die Roboter drehen sich drei Mal um die eigene Achse. Die Rotation über die Zeit von allen vier Robotern wird dargestellt. Seine Position verändern die Roboter dabei nur marginal.*

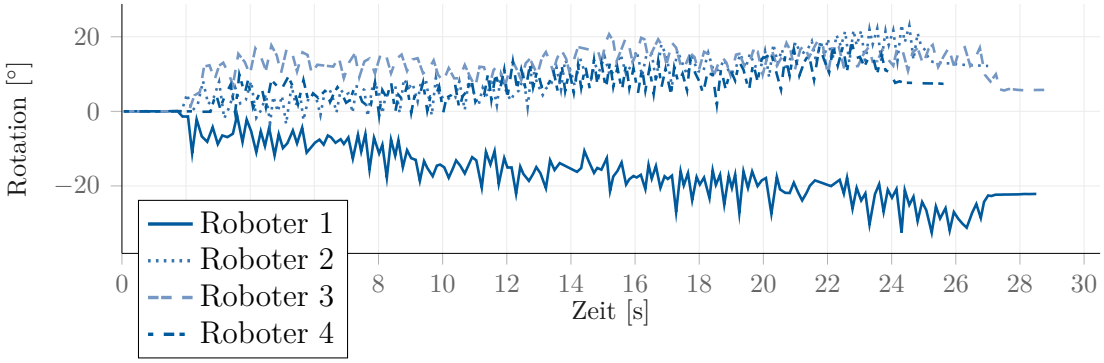

Abbildung 7.6: *Die Abweichung der berechneten zur tatsächlichen Rotation für die vier Roboter ist in der Abbildung dargestellt. Im Gegensatz zur Drehung nach links fällt die Abweichung deutlich geringer aus.*

7.3 Experiment III: Folgen eines Balles

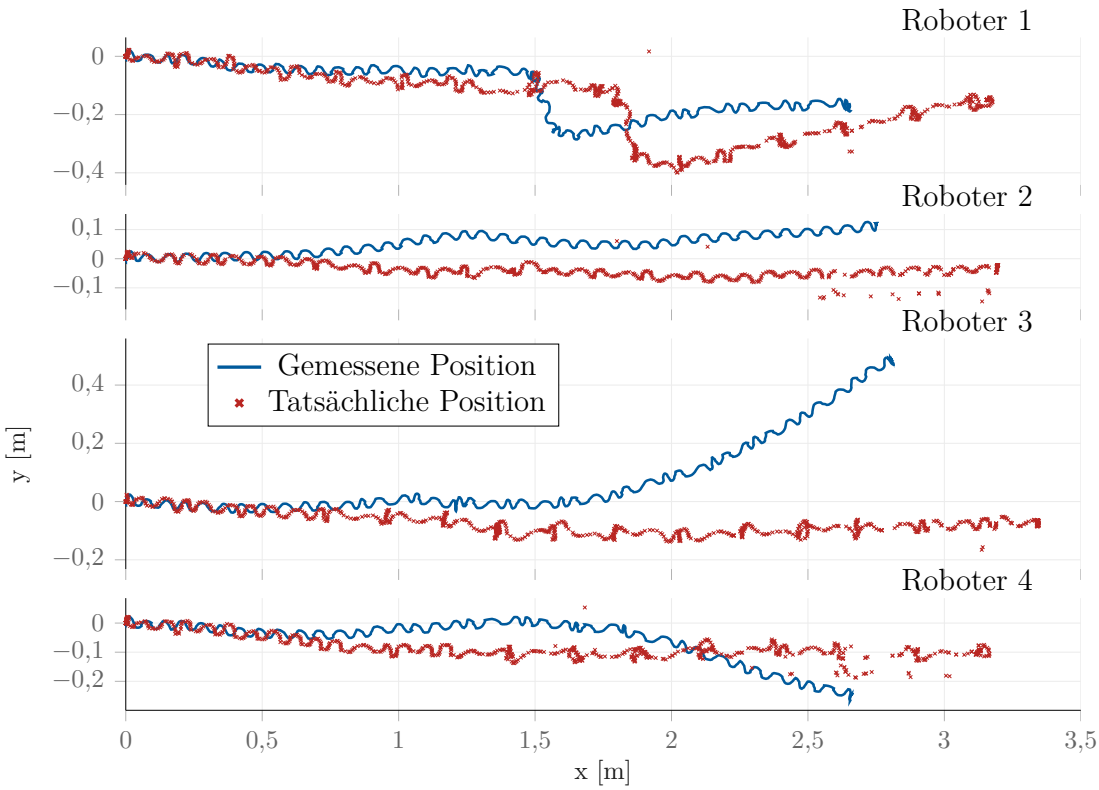

Abbildung 7.7: *Die Roboter folgen einem Ball der vor ihnen liegt und probieren sich diesem anzunähern. Das entsprechende Verhalten wird in üblichen Spielsituationen im RoboCup-Kontext verwendet. Das heißt, dass der Roboter sich während des Laufs auch umschaut. Der erste Roboter erkennt dabei einen anderen Ball und verändert seine Trajektorie maßgeblich. Die Graphik zeigt die Positionen, die die Roboter bei diesem Experiment tatsächlich haben und die, die sie berechnen.*

Im Gegensatz zu den vorhergehenden Experimenten sollen vier Roboter keine konstante Bewegung ausführen. Viel mehr sollen sie den dynamischen Lauf nutzen, wie er auch in einem Spielszenario verwendet wird, und einem Ball geradlinig folgen.

Das erste Experiment hat gezeigt, dass die Roboter keineswegs geradeaus laufen, wenn sie dies als Auftrag haben. Demnach stellt dieses Experiment bereits eine weitere Komplexitätsstufe dar.

Die Abbildung 7.7 zeigt den Verlauf des Experiments. Der rot markierte Graph zeigt die tatsächliche Position, der blaue Graph die selbst ermittelte Position der Roboter.

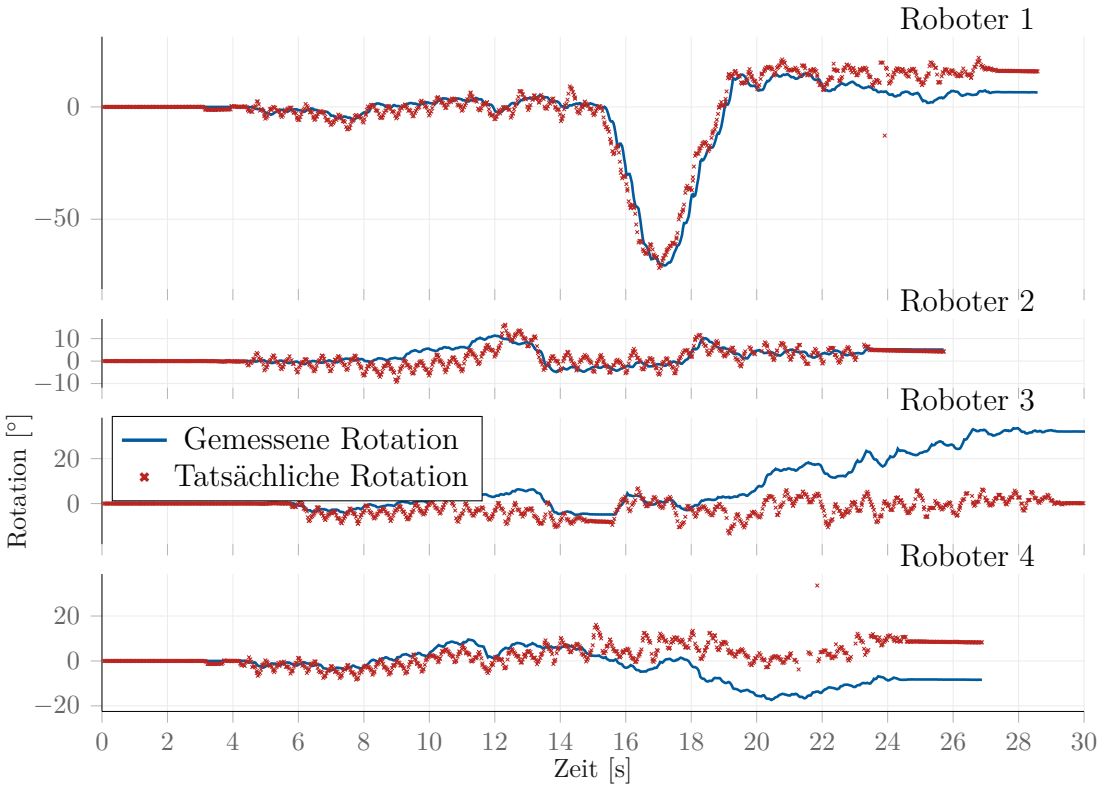

Abbildung 7.8: *Die Graphiken zeigen die entsprechende Rotation, die die Roboter während des Experiments berechnen. Der zweite Roboter misst diese Rotation sehr akkurat.*

Alle Roboter haben es erneut gemein, dass sie ihren tatsächlichen Weg nach vorne unterschätzen. Dies bestätigt die Beobachtung aus dem Experiment 7.1, dass es sich um einen systematischen Fehler handeln muss.

In Abbildung 7.8 ist die Rotation der Roboter während des Experiments zu dargestellt. Insbesondere für den zweiten Roboter ist diese äußerst akkurat. Der erste Roboter ist in der Lage, seine kurzzeitige Fehleinschätzung sehr gut zu berechnen.

sind die einzelnen Positionskomponenten aufgeschlüsselt. Im Gegensatz zum ersten Experiment ist in diesem Versuch zu erkennen, dass die Positionsdifferenz im Wesentlichen durch die y-Komponente hervorgerufen wird. Der Anstieg der Differenz von tatsächlicher und ermittelter Position ist dabei größer als im Vergleich zum ersten Experiment. Der absolute Fehler der Positionsermittlung durch den Roboter ist nach 30 Sekunden beinahe doppelt so hoch.

Das Experiment zeigt deutlich, dass durch das Hinzufügen von Komplexität in der Aufgabenstellung, die Messungsungenauigkeit ansteigt. Sobald der Roboter das dynamische Anpassen seiner Schritte nutzt, steigt also der Fehler an. In nachfolgenden Experimenten wird dieser Aussage noch mehr Ausdruck verliehen.

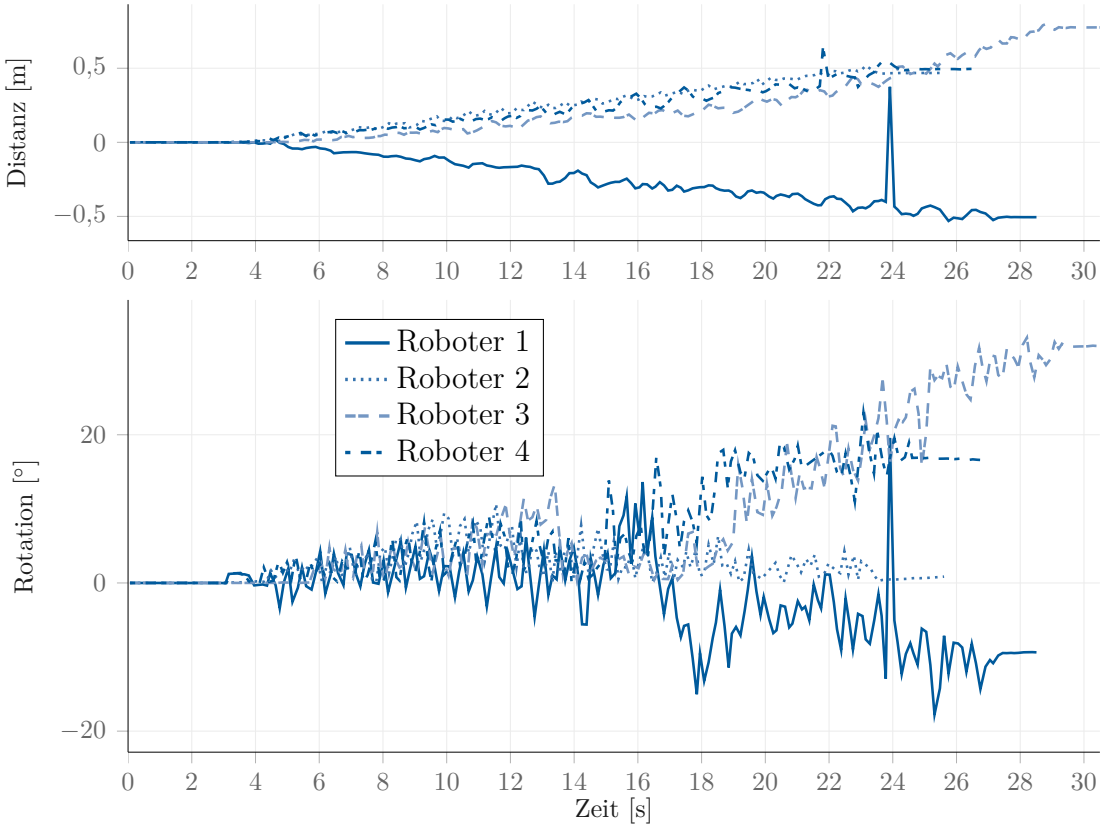

Abbildung 7.9: *In der oberen Abbildung ist die Distanz zwischen der gemessenen und tatsächlichen Position abgetragen. Die untere Grafik zeigt die Abweichung der Rotation. Beide Abweichungen akkumulieren über die Zeit einen Fehler.*

7.4 Experiment IV: Kombination Drehung und Geradeauslaufen

Für den Roboter entspricht das Laufen eines Kreises dem Geradeauslaufen mit einer festgeschriebenen Fußdrehung.

Zur Illustration der Fehlerauswirkungen, wird in diesem Experiment anhand eines Roboters die Abweichung ermittelt. In Abbildung 7.10 ist der Verlauf des Experiments zu erkennen. Drei Teilstücke des Experiments wurden herausgezogen und dargestellt. Zwischen der zweiten und dritten Abbildung ist ein zeitliche Lücke der übersichthalber eingefügt. Der Roboter soll exakt drei Kreise laufen. Die tatsächliche Position (Ground Truth) zeigt, dass der Roboter in der Lage ist, nahezu exakt diese drei Kreise zu laufen. Für das Laufen dieser drei Kreise braucht der Roboter 53 Sekunden.

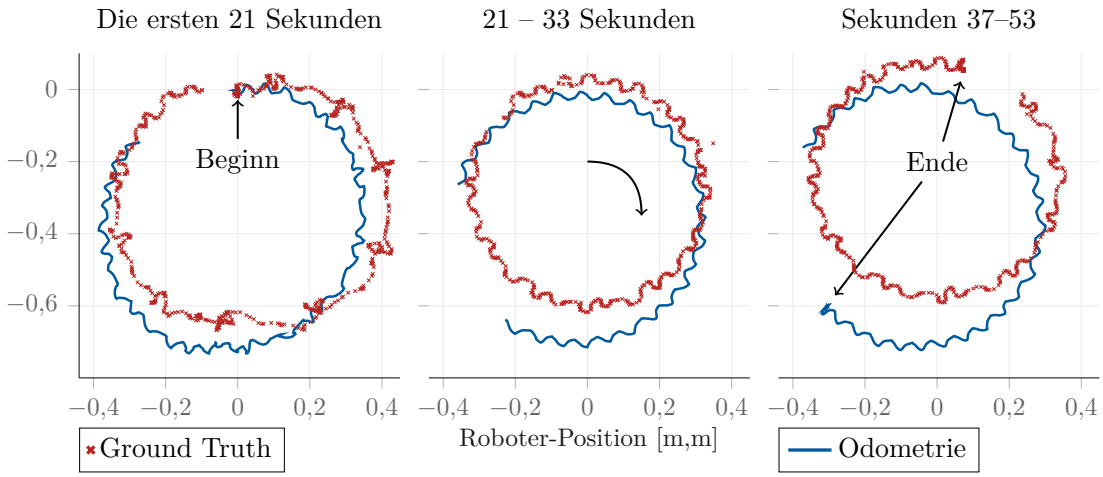

Abbildung 7.10: *Die abgelaufenen Koordinaten des Roboters, während er drei Kreise abläuft.*

Die Abbildung 7.11 stellt die einzelnen Komponenten der Roboterpose in Abhängigkeit der Zeit dar. Der Fehlerverlauf der x- und y-Komponente ist dabei sehr ähnlich. Die absoluten Werte übersteigen allerdings die Messungsungenauigkeiten des ersten Experiments. Auch der Rotationsfehler ist deutlich ausgeprägter gegenüber dem reinen Rotieren um die eigene Achse aus dem zweiten Experiment.

Die absoluten Messungsungenauigkeiten der Distanz und der Rotation können der Abbildung 7.12 entnommen werden. Es ist zu erkennen, dass sich sowohl der Rotationsfehler, als auch der Distanzfehler über die Zeit wie erwartet akkumuliert. Beide Fehler sind deutlich ausgeprägter als in den abgeschlossenen vorhergehenden Experimenten.

Es kann daher vermutet werden, dass der Fehler umso größer wird, je mehr Achsen in den Lauf berücksichtigt werden.

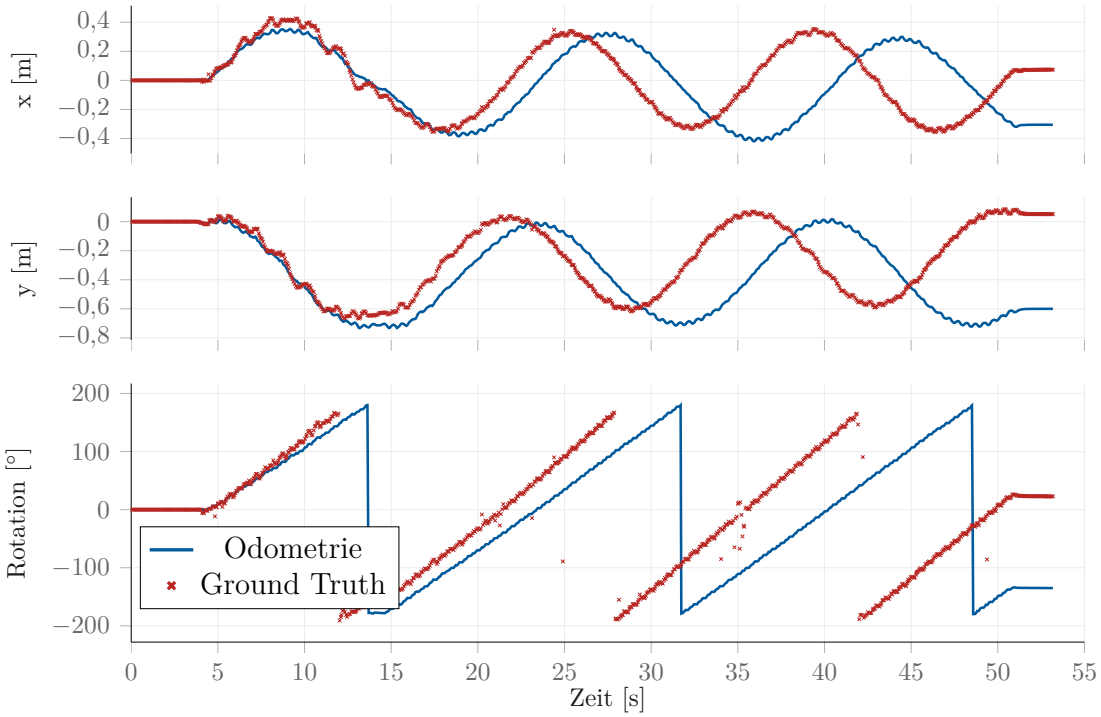

Abbildung 7.11: *Die Komponenten x, y und Rotation in Abhängigkeit der Zeit einzeln aufgegliedert.*

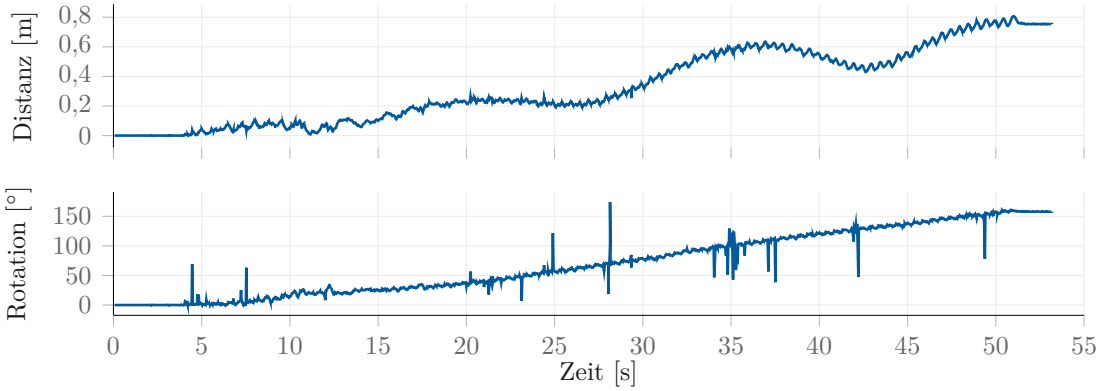

Abbildung 7.12: *Die Graphik zeigt die Differenzen für tatsächliche und ermittelte Daten der Position und Rotation. Gegenüber den beiden isolierten ersten Experimenten ist zu erkennen, dass sowohl die Differenz der Rotationen, als auch die Komponenten der Positionen deutlich höheren Messunterschieden ausgesetzt sind. Die Kombination verschiedener Laufstile führt demnach zu einer Erhöhung des Messunterschiede.*

7.5 Experiment V: Folgen eines Balles mit komplexen Bewegungswechseln

Zusätzlich zum vorhergehenden Experiment erhöht sich die Komplexität für den Roboter erneut. Der Roboter soll einem Ball folgen, dabei sind mehrfache Richtungsänderungen mit inbegriffen. Die Abbildung 7.13 zeigt den Verlauf der Position des Roboters während der Durchführung des Experiments. Die Trajektorie des Roboters ist dabei *achtförmig*.

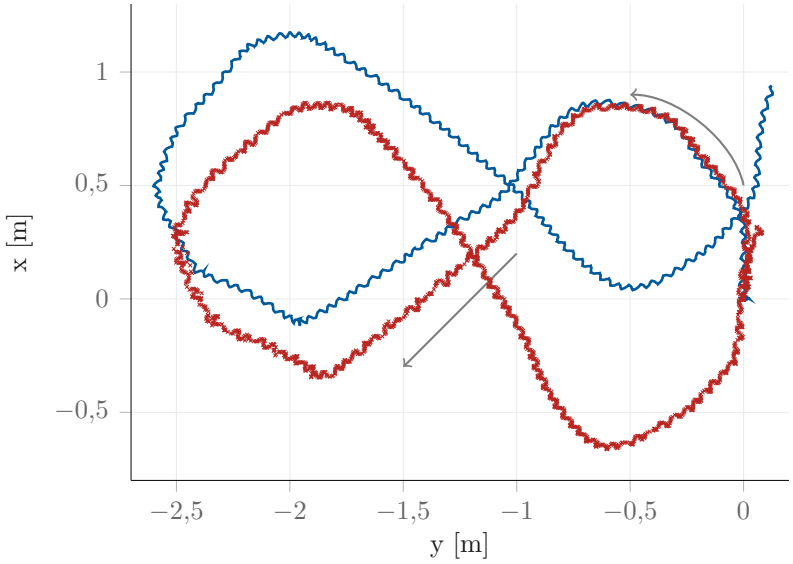

Abbildung 7.13: *Der Roboter läuft eine komplexe Trajektorie mit vielen Bewegungsänderung ab. Dabei nutzt er sein Verhalten aus einem typischen Spiel und folgt einem Ball, den man vor ihm herführt, so dass ein achtförmiges Muster entsteht. Viele Richtigungswechsel sind dabei auszumachen.*

Die Graphen der Abbildung 7.14 zeigen dabei den Verlauf der x- und y-Komponente, sowie die Distanz aus der gemessenen und tatsächlichen Position.

Es ist zu erkennen, dass auch hier, wie in allen vorhergehenden Experimenten, die Distanz aus der gemessenen und tatsächlichen Position ansteigt.

Positiv ist zu erkennen, dass die Falschberechnung des Roboters nach einer Minute konstantem Laufen lediglich 70 cm beträgt.

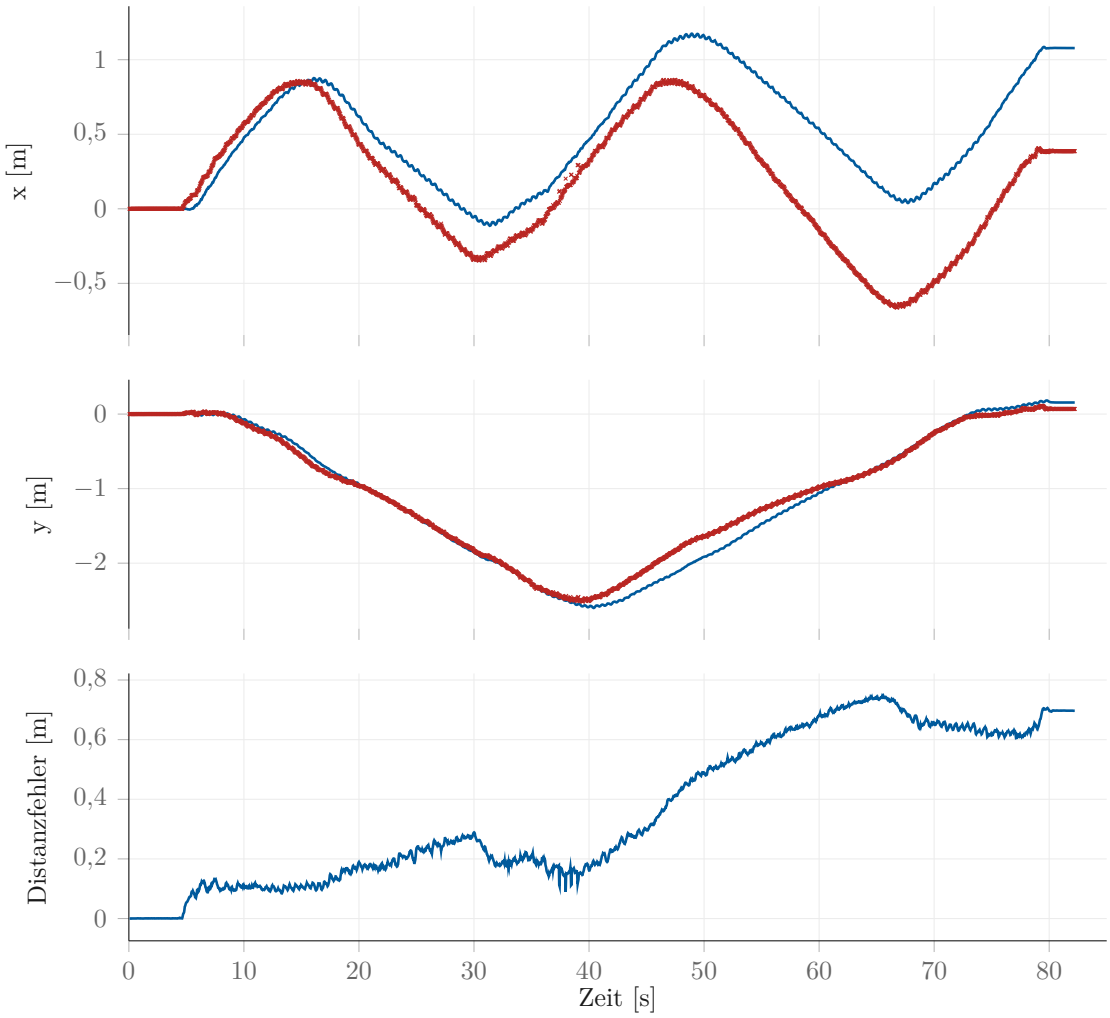

Abbildung 7.14: *Beim Ablaufen einer achtförmigen Trajektorie akkumuliert sich, wie zu erwarten war, ebenfalls ein Fehler in der Odometrie. Die Graphik zeigt die x- und y-Komponenten dieses Vorgangs. Die untere Graphik zeigt die Distanzdifferenz zwischen Odometrie und tatsächlicher Position. Trotz der vielen einbegriffenen Richtungswechsel und dynamischen Laufkomponenten ist der Fehleranstieg nicht viel höher als in vorhergehenden Experimenten. Dies kann in der symmetrischen Struktur begründet liegen.*

7.6 Experiment VI: Spielszenario mit Zweikampfverhalten

Ein wesentlicher Bestandteil eines Fußballspiels besteht in der Balleroberung bzw. Ballbehauptung. Dies bedeutet, dass in einem sogenannten *Zweikampf* der Ball

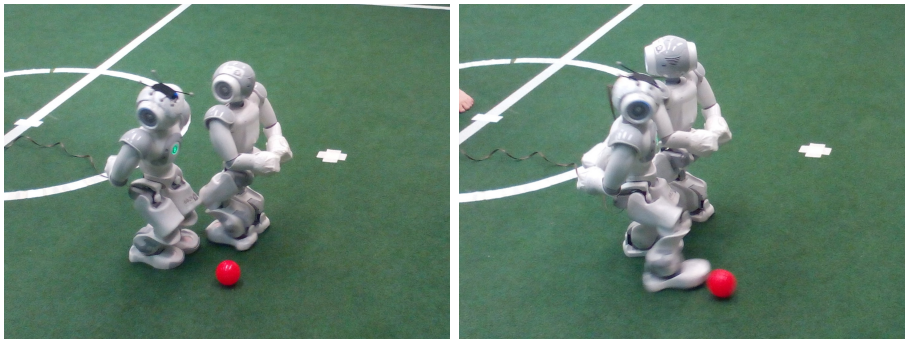

Abbildung 7.15: *Die Abbildung zeigt das Zweikampfverhalten aus diesem Experiment. Der angreifende Roboter war in der Lage, den Ball am verteidigenden Roboter vorbeizuführen. Durch Seitwärtsbewegungen entzieht sich der Roboter dem Einfluss des angreifenden Roboters, wird aber beim Vorbeilaufen wesentlich verschoben.*

von einem anderen Roboter erobert werden soll. Andersherum gilt für angreifende Roboter, dass sie den Ball gegenüber verteidigenden Robotern behaupten sollen.

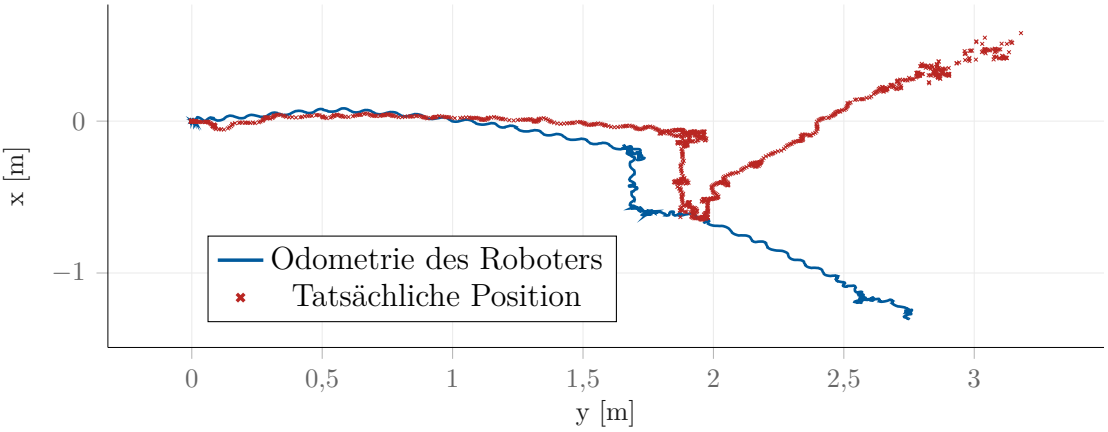

Abbildung 7.16: *Diese Graphik zeigt die gemessene und tatsächliche Position des Roboters während des Zweikampf-Experiments. Das Tor befindet sich dabei oben rechts. Es ist zu erkennen, dass der Roboter nach mehreren Seitwärtsschritten seine Positionsvorstellung völlig falsch ist. Der Roboter wurde während des Zweikampfes um seine $z-$Achse gedreht.*

Diese Situation sind innerhalb eines Spieles sehr häufig. Dieses Experiment zeigt einen Zweikampf zweier Roboter des Nao Team Humboldts. Das Experiment soll potenzielle Schwierigkeiten mit dem Umgang zweier Roboter im Zweikampfverhalten aufzeigen.

Das Experiment wurde wie folgt durchgeführt. Zwei Roboter werden gestartet, ein Ball liegt von beiden Robotern aus gesehen etwa gleich weit entfernt. Beide Roboter probieren in einem Zweikampf den Ball zu behaupten. Ohne Beschränkung der Allgemeinheit gilt der zu untersuchende Roboter als angreifender Roboter. Die Situation während des Zweikampfes wird in Abbildung 7.15 dargestellt. Dem angreifenden Roboter gelingt es, den Ball am verteidigenden Roboter vorbei zu spielen. Daraufhin versucht der angreifende Roboter durch Seitwärtsschritte am

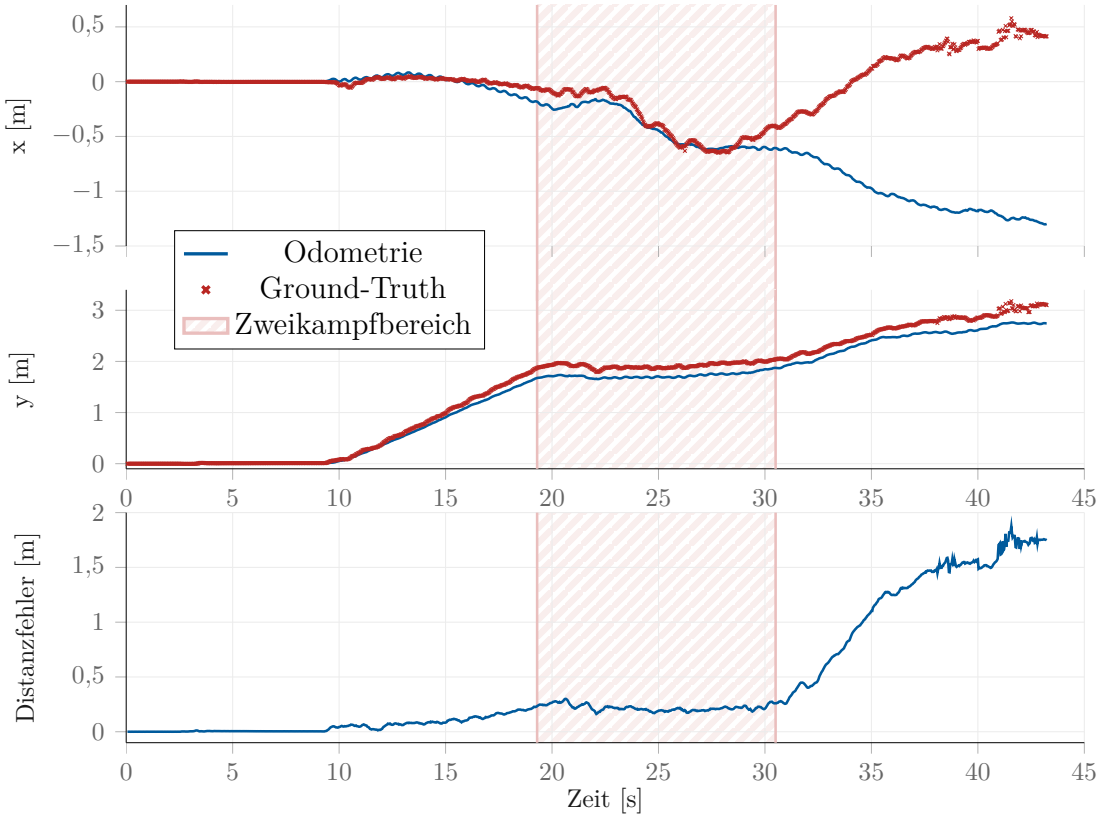

Abbildung 7.17: *Diese Graphik zeigt die einzelnen Komponenten der gemessenen und tatsächlichen Position des Roboters, sowie der sich entwickelnde Fehler zwischen den beiden Positionen. Es ist zu erkennen, dass nach dem Zweikampf der Roboter (rot schraffierte Bereich) der Fehler sehr stark ansteigt. Dies liegt darin begründet, dass der Roboter während des Zweikampfes um seine $z-$ Achse verschoben wurde.*

verteidigenden Roboter vorbeizulaufen, um danach den Ball erreichen zu können.

Der verteidigende Roboter hat den Ball aus dem Sichtfeld verloren und sucht den Ball, bei seiner Drehung lenkt er den angreifenden Roboter entscheidend ab. Der angreifende Roboter wird dabei um seine Z-Achse verschoben, kann aber den Ball behaupten und sich weiter Richtung Tor fortbewegen. Die Graphik zum Positionsverlauf ist in der Abbildung 7.16 dargestellt. In dieser Abbildung ist zu erkennen, wie der angreifende Roboter in einer bestimmten Situation durch Seitwärtsschritte dem verteidigenden Roboter *umlaufen möchte*. Dies gelingt, aber kurz darauf ist seine Vorstellung von seiner Position stark rotiert. Dies bestätigt die vorhergehende Beschreibung, dass der Roboter während des Zweikampfes um seine Z-Achse rotiert worden ist.

Darüber hinaus ist zu erkennen, dass während der Zweikampfphase der Fehler hingegen kaum verändert. Dies liegt hauptsächlich darin begründet, dass die Positionsänderung in dieser Phase sehr gering ist.

Eine konkrete Aussage experimentell zu ermitteln ist äußerst schwierig. Jede Mannschaft im RoboCup hat andere Strategien, den Zweikampf anzugehen. Zur Minimierung der Kontaktmöglichkeiten verschränen die Roboter des Nao Team Humboldts beispielsweise ihre Arme permanent nach hinten. Dieses Gehverhalten verhindert unbeabsichtigtes *Hängenbleiben* des Roboters mit seinen Armen. Beobachtungen zeigen, dass Delokalisierungen nach einem Zweikampf so verringert werden konnten.

7.7 Zusammenfassung

Ein Fußballspiel zeichnet sich durch eine sich schnell verändernde Umwelt aus. Ein dynamische Anpassung der Lauftrajektorie ist unabdingbar, um dieser schnellen Umweltänderung gerecht zu werden. Die Experimente zeigen, dass sich die Messungenauigkeit mit der dynamischen Anpassung des Laufes an die sich ändernden Umweltgegebenheiten steigern.

Darüber hinaus zeigen die Experimente, dass der Roboter das Wissen über seine eigene Position nicht ausschließlich auf die Berechnungen der Odometrie stützen kann. Eine Halbzeit eines RoboCup-Spiels dauert 10 Minuten. Bereits nach einer Minute, abhängig von der Bewegung, ist der Roboter ohne Fremdeinwirkung um über einen halben Meter versetzt.

Ein Anstaz die Odometrie zu verbessern ist das Integrieren von visuellen Informationen. *Visuelle Odometrie* wird bereits vielfach angewendet. Im wesentlichen basiert die Technik auf zwei Schritten. Zunächst werden interessante Punkte (*feature Points*) im Bild erkannt. Der zweite Schritt ist es, diese Punkte in allen Kameras aufzufinden (*Stereo Vision*)[1] und auf aufeinanderfolgende Bildern. Aus der

[1] Der Roboter *Nao* hat keine ausreichend überlappenden Kameras, so das dieser Schritt nicht in Betracht gezogen wird.

Veränderung des Punktes können Rückschlüsse auf die Bewegung der Kamera geschlossen werden. Besonders bieten sich für diesen Zweck RGB-D-Kameras an, da sie durch die Information der Punkttiefe den zweiten Schritt deutlich vereinfachen.

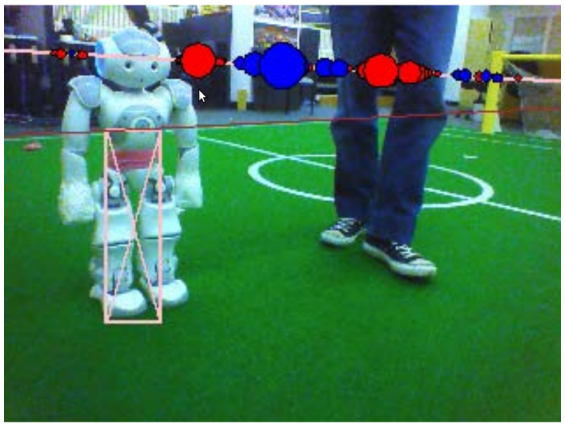

Abbildung 7.18: *Aus der Veröffentlichung [AH14]. Entlang eines Horizontbandes des Roboters wird SURF ausgeführt. Aufgrund der Annahme, dass der Roboter sich nur planar bewegt, ist dies äußerst kosteneffizeient Die extrahierten* feature *werden genutzt um aufeinanderfolgende* Frames *zu vergleichen und die Bewegung des Roboters zu ermitteln.* Feature *erkannter Roboter werden dabei ausgespart.*

Vielversprechende Ansätze zur Verbesserungen der Odometrieinformationen wurden bereits durch andere RoboCup-Teams eingeführt. Im Kapitel 4 habe ich bereits darauf Bezug genommen. *Guerrero* und *Ruiz-del-Solar* schlugen 2007 vor, temporäre Landmarken in die Lokalisation mit einzubeziehen. Sie nutzen dafür Informationen, die sie als räumlich und zeitlich *lokale* Informationen bezeichnen. Das Nutzen dieser lokalen Informationen soll helfen, den Odometrie-Fehler abzuschätzen und somit die globale Lokalisierung zu verbessern [GS08].

In der Veröffentlichung *Fast Monocular Visual Compass for a Computationally Limited Robot* von *Anderson* und *Hengst* wird ein weiter Ansatz zur visuellen Odometrieberechnung anhand von *Feature Points* geliefert [AH14]. Dabei werden besondere *Feature* eines Bildes lediglich entlang des Horizonts des Roboters gesucht. Andere Bereiche bleiben ausgespart, da die Annahme getroffen wird, dass sich der Roboter in einer ebenen Umgebung fortbewegt und vertikale Bewegungsänderung in dieser Umgebung nicht vonstatten gehen. Anders ausgedrückt, dynamische Objekte der Umgebung wie andere Roboter, Schiedsrichter oder Zuschauer, bewegen sich alle auf einer Ebene. Gefundene *Feature* werden um bekannte unabhängige dynamische Objekte reduziert. In Abbildung 7.18 ist dieser Schritt zu erkennen. Die *Feature* eines erkannten Roboters werden nicht berücksichtig.

Betrachtet man alle Frames einer Videosequenz kontinuierlich, so wird auch bei der visuellen Odometrie ebenfalls eine akkumulierter Fehler, ein Drift, entstehen. Um dies zu umgehen, bedient man sich einer *Sliding-Window*-Technik. Man betrachtet lediglich die letzten drei *Frames* und vergleich diese mit dem ersten. *Frames* die dabei das Ergebnis deutlich verändern, werden verworfen [AH14].

Eine zusätzliche Fehlerquelle zeigt das letzte Experiment auf. Unabhängig von eigener Berechnungen, kann der Roboter während eines Zweikampfes mit einem

anderen Roboter verschoben werden. Dies korrespondiert nicht mit seiner Odometrie. Es ist denkbar, dass hier visuelle Odometrie aus [GS08, AH14] abhilfe schaffen kann. Eine Schwierigkeit besteht jedoch darin, wenn zwei Roboter besonders nah beiander stehen, und dem anderen Roboter die Sicht verstellen. Dies ist ein sehr typisches Szenario. Die Methode der visuellen Odometrie müsste in diesem Punkt genauer untersucht werden. Um diese Situation klar besser deuten zu können, müsste das Gyrometer des Roboters mit einer zusätzlichen z-Achse ausgestattet werden. Andernfalls ist die Vorhersagung der Eigenbewegung (Bewegungsmodell) nicht akkurat möglich. Im Allgemeinen bleibt jedoch anzumerken, dass die aktuelle Odometrieberechnung für kurze zeitliche Rückblicke ein sehr gutes Bewegungsmodell liefert.

8 Empirische Analyse des Multi-Hypothesen-Tormodells

Dieses Kapitel untersucht empirisch die Implementation des Multi-Hypothesen-Tormodells (MHTM) aus Kapitel 5. In verschiedenen Experimenten wird analysiert, wie gut das MHTM die im Abschnitt 1.2 formulierten Anforderungen erfüllt. Insbesondere geht es um den Umgang mit folgenden Arten von Unsicherheit:

- Sensorrauschen aus der Bildverarbeitung und der Kinematik
- Spärliche Falschmessungen *(false-positives)*
- Systematischen Inkonsistenzen resultierend aus dichten Falschmessungen

Man beachte dabei die Unterteilung der Falschmessungen in *spärlich* und *dicht*. Eine Fehlmessung kann recht einfach dadurch erkannt werden, dass sie nicht konsistent mit dem bestehenden Modell ist. Solche Messungen werden im Perzeptpuffer abgefangen. . Im Gegensatz zu spärlichen Fehlmessungen, sind die dichten Fehlmessungen dadurch charakterisiert, dass sie oft und konsistent genug auftreten, dass sich ein kohärentes Cluster im Perzeptpuffer bilden kann. Aus diesem Cluster wird daraufhin eine neue Multi-Hypothese im MHTM erzeugt.

Erfahrungen haben gezeigt, dass ein Torpfosten zuverlässig detektiert werden kann, falls dieser im Sichtbereich des Roboters ist. Das Feld selbst folgt zwar hinsichtlich der Dimensionen und Farben vorgeschriebenen Regeln (vgl. 2.1), die Umgebung ist jedoch nicht geregelt. Daher kommt es immer wieder vor, dass fälschlicherweise ein Torpfosten erkannt wird. Beispielhaft seien hier Werbebanden erwähnt, aber auch das Hosenbein eines Zuschauers direkt am Feldrand kann dazu führen. In der Regel treten solche Artefakte sporadisch (spärlich) als Ausreißer auf. Solche spärlichen Falschmessungen werden von dem *Puffersystem* abgefangen. Im Experiment 8.3 wird diese Art von Falschmessungen genauer untersucht.

Es kann nicht ausgeschlossen werden, dass es auch systematische (dichte) Falschmessungen gibt. Diese Falschmessungen treten regelmäßig auf und führen dazu, dass eine neue Multi-Hypothese im MHTM erzeugt wird. Das könnte etwa bei schlechter Beleuchtung der Fall sein, wenn die Torfarbe nicht stark genug von der Umgebung zu trennen ist und Objekte am Feldrand systematisch fälschlicherweise als Torpfosten erkannt werden.

Unsicherheiten	Methoden des Algorithmus
Sensorrauschen	Partikelfilter
Spärliche Fehlmessungen	Konsistenzprüfung (Assoziation, Perzeptpuffer)
Dichte Fehlmessungen	Extraktion des Tormodells, semantische Interpretation der Multi-Hypothesen

Tabelle 8.1: *Gegenüberstellung verschiedener Unsicherheiten in den Sensordaten und entsprechender Werkzeuge des MHTM zu deren Behandlung.*

Diese falschen Torpfosten können nur im Zusammenhang mit anderen (richtigen) Pfosten während der Extraktion des Tormodells detektiert werden. Das entsprechende Verhalten von MHTM im Fall systematischer Falschmessungen wird im Experiment 8.4 untersucht.

Diese Unsicherheiten werden im MHTM - Algorithmus auf verschiedenen Ebenen entsprechend behandelt wie in der Tabelle 8.1 zusammengefasst. Die Assoziierungsmethode des Algorithmus und das damit verbundene Vorhalten von Perzepten ist die Instanz, um spärliche Falschmessungen zu verwerfen. Perzepte, die zu keinem Filter passen, werden in einem Perzeptpuffer gespeichert. Nur bei genügend vielen Perzepten wird daraus eine Multi-Hypothese erzeugt. Das Sensorrauschen (Falschwahrnehmung der Pfosten) wird von Partikelfiltern verarbeitet. Systematische Inkonsistenzen führen zur Bildung neuer Multi-Hypothesen, da sie einen potenziellen Pfosten darstellen. Die Extraktion des Tormodells ist dafür verantwortlich, dass ein angemessenes Tor aus den aktuellen Multi-Hypothesen extrahiert wird.

Erläuterung zum Aufbau des Abschnitts

Jeder der in der Tabelle 8.1 zusammengefassten Aspekte wird in den folgenden Experimenten exemplarisch untersucht. Ohne Beschränkung der Allgemeinheit wird dabei zur Analyse der linke Torpfostens gewählt. Die Abschnitte 8.1 und 8.2 untersuchen den Einfluss des Sensorrauschens des kinematischen Modells und der visuellen Verarbeitung auf die einzelnen Multi-Hypothesen (Partikelfilter). Im Abschnitt 8.3 werden spärliche Falschmessungen simuliert und die Robustheit der Konsistenzprüfung des MHTM auf der Perzeptebene getestet. Der Abschnitt 8.4 beschäftigt sich mit dem Problem dichter Falschmessungen. Abschließend wird im Abschnitt 8.5 ein dynamisches Experiment durchgeführt und die Robustheit des MHTMs in Bewegung untersucht.

8.1 Experiment I: Analyse des stationären „Gehens“

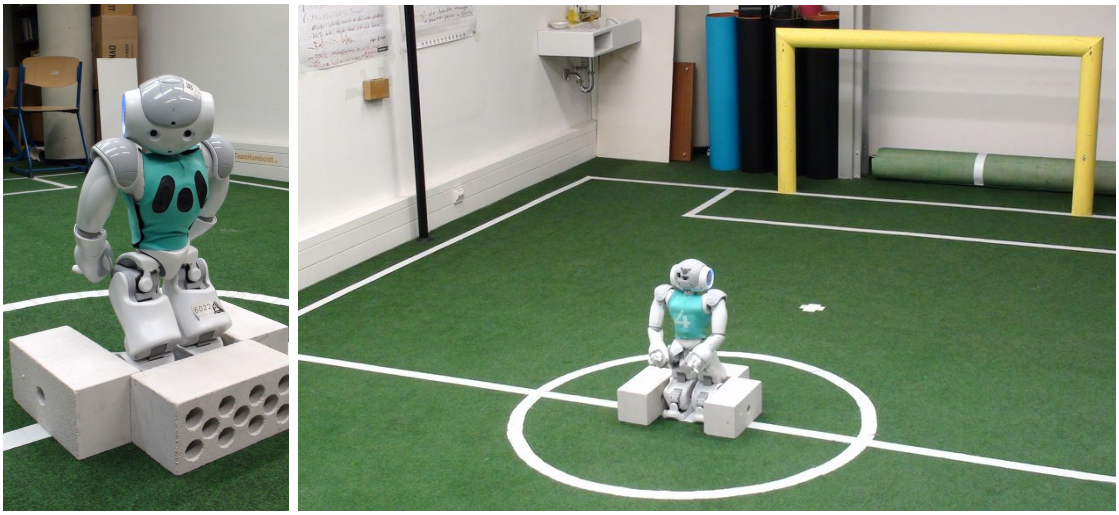

Abbildung 8.1: *Der Roboter befindet sich im Zentrum des Spielfelds und tritt auf der Stelle. Er schaut dabei nach vorne direkt auf das Tor.*

In diesem Experiment wird der Einfluss des Rauschens in der Kinematik des Roboters auf das MHTM untersucht. Dazu befindet sich der Roboter im Koordinatenursprung des Felds und *läuft* auf der Stelle. Er schaut dabei nach vorne direkt auf das Tor. Die Fehler werden durch das Laufen hervorgerufen. Da sich der Roboter aufgrund der Bewegungsfehler beim Tippeln ungewollt langsam bewegt, wird seine Position durch eine spezielle Vorrichtung fixiert. Diese ist in der Abbildung 8.1 dargestellt.

Die Abbildung 8.2 veranschaulicht die aufgenommenen Perzepte im Verlauf des Experiments (links) und eine Momentaufnahme erzeugter Multi-Hypothesen der Torpfosten am Ende des Experiments (rechts). Diese Gegenüberstellung ist zulässig, da entsprechend der Markow-Eigenschaft (vgl. 3.2) alle Zustände der Multi-Hypothesen des Experimentverlaufs in dieser Momentaufnahme enthalten sind.

Die Multi-Hypothesen sind durch die darunterliegende Partikelwolken und die jeweiligen Kovarianz-Ellipsen dargestellt. Der schwarze Balken visualisiert das gebildete Tormodell. Mit Hilfe des Modells lassen sich die einzelnen Multi-Hypothesen den entsprechenden Torpfosten zuordnen. Diese Zuordnung wird durch die Farbe der Partikel visualisiert, dabei steht rot für den linken und blau für den rechten Torpfosten. Der Roboter sieht die beiden Pfosten des Tors permanent. Daher stellt die Zuordnung der Perzepte und die Erzeugung des Tormodells kein Problem dar. Die oberen zwei Graphen der Abbildung 8.3 zeigen den Winkel und die Distanz der Perzepte, die dem linken Torpfosten zugeordnet wurden (grüne Punkte) im

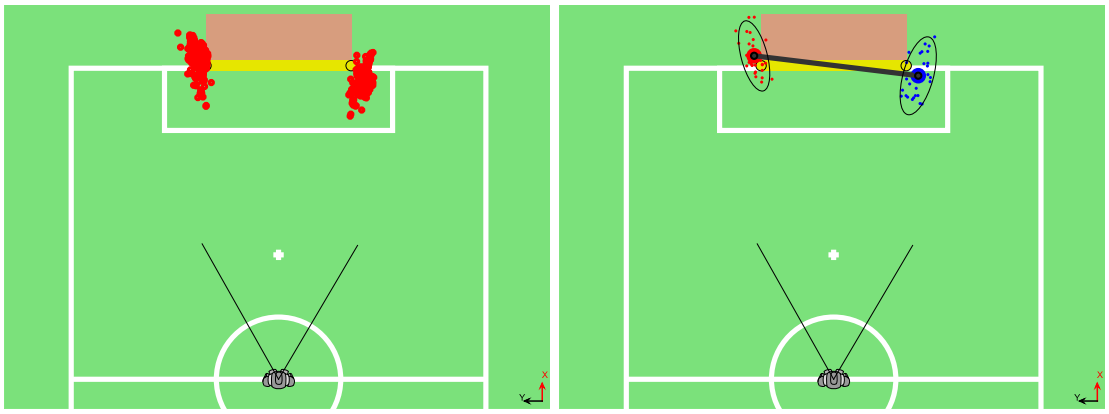

Abbildung 8.2: *Der Roboter befindet sich im Zentrum des Spielfelds, tritt auf der Stelle und schaut dabei nach vorne direkt auf das Tor. Die linke Abbildung veranschaulicht die Torpfostenperzepte als rote Kreise. Diese wurden über den gesamten Verlauf des Experiments gesammelt. Das rechte Bild zeigt eine Momentaufnahme des MHTM am Ende des Experiments. Die einzelnen Multi-Hypothesen werden durch die entsprechenden Partikelwolken und die jeweilige Kovarianz-Ellipsen dargestellt, dabei veranschaulicht die Farbe die Klassifizierung der Torpfosten (rot - linker Torpfosten, blau - rechter Torpfosten). Der schwarze Balken illustriert das extrahierte Tormodell.*

Vergleich zu dem daraus erzeugten Modell des linken Torpfostens (blaue Linie), das dem Erwartungswert des entsprechenden Partikelfilters entspricht. Beides ist über die gesamte Laufzeit des Experiments abgetragen. Besonders gut zu erkennen ist die Filterleistung des Partikelfilters: Das erzeugte Modell des Torpfostens hat eine deutlich niedrigere Varianz als die Perzepte. Dies gilt sowohl für den Winkel, als auch für die Distanz. Der untere Graph der Abbildung 8.3 zeigt die Varianz in der Richtung beider Hauptachsen der Partikelwolke. Es ist gut zu sehen, dass die Ausdehnung des Filters den Radius von 20 cm entlang der langen Hauptachse nicht überschreitet. Ebenfalls gut zu sehen ist der Größenunterschied beider Achsen. Im Abschnitt 6.1 wurde empirisch analysiert, dass die Distanzmessungen deutlich verrauschter sind als die horizontalen Winkelmessungen. Die Größenunterschiede der Hauptachsen sind dadurch zu erklären.

Zusammengefasst demonstriert dieses Experiment die Grundfunktionalität des MHTM in einem einfachen Szenario. Wir können schlussfolgern, dass die zugrundeliegenden Partikelfilter sehr gut in der Lage sind, das Rauschen des kinematischen Modells zu handhaben.

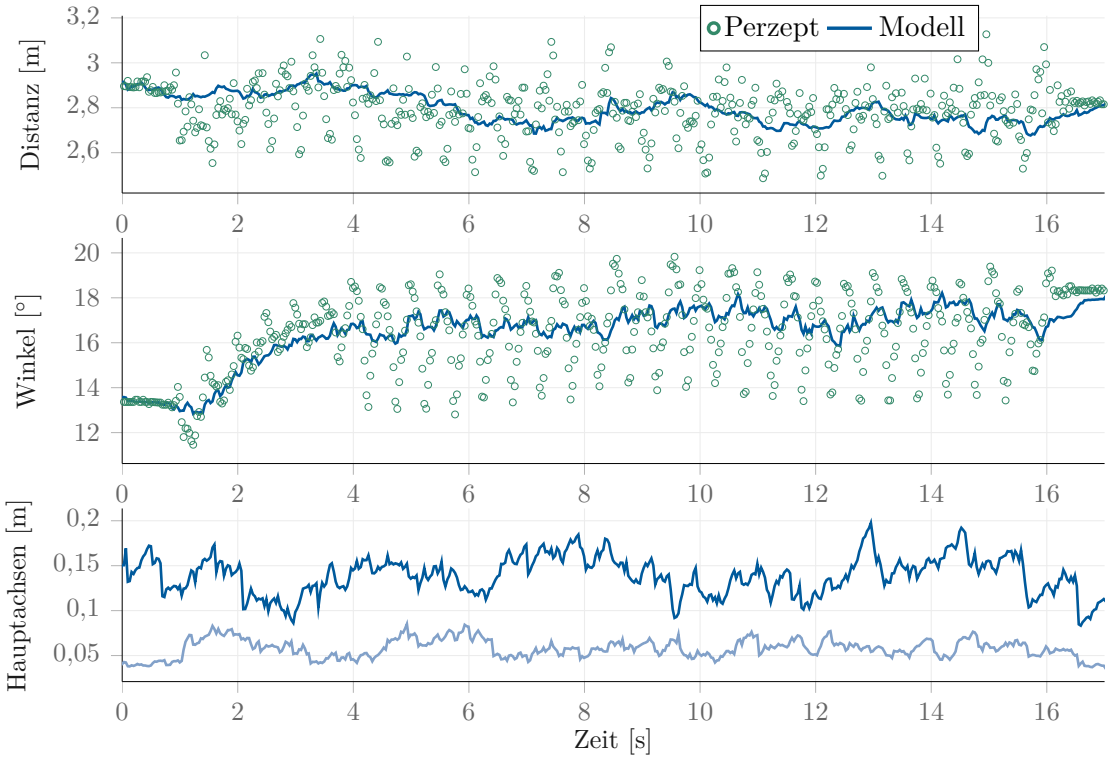

Abbildung 8.3: *Die beiden oberen Graphen zeigen die Perzepte die mit dem linken Pfosten des Modells assoziiert werden. Der obere Graph vergleicht die Distanzen dieser Perzepte und dem Modell, der Mittlere den entsprechenden Winkel. Die untere Abbildung zeigt die Länge der beiden Hauptachsen der Partikelwolke der Multi-Hypothese, die den linken Pfosten modelliert.*

8.2 Experiment II: Analyse bei Suchverhalten

In diesem Experiment wird das Verhalten des MHTMs bzgl. einer Kopfbewegung des Roboters untersucht. Der Fokus liegt hierbei insbesondere auf der unterbrochenen Wahrnehmung der Torpfosten. Dazu befindet sich der Roboter im Zentrum des Feldes ausgerichtet in Richtung des gegnerischen Tors. Abbildung 8.1 zeigt den Versuchsaufbau. Der Roboter bewegt dabei nur seinen Kopf mehrmals von links nach rechts. Die Suchbewegung entspricht dabei exakt der Bewegung, die der Roboter im Spiel zur Exploration seiner Umgebung anwendet. Während der Suche verliert der Roboter systematisch die Torpfosten aus dem Blick.

Die Abbildung 8.4 veranschaulicht die Perzepte aufgenommen im Verlauf des Experiments (links) und eine Momentaufnahme erzeugter Multi-Hypothesen der Torpfosten am Ende des Experiments (rechts). Die Multi-Hypothesen sind durch die darunterliegende Partikelwolken und die jeweiligen Kovarianz-Ellipsen darge-

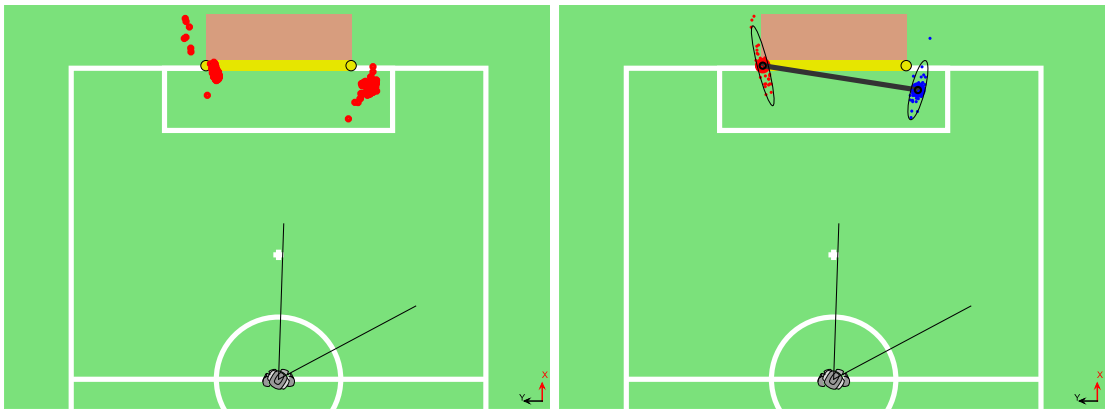

Abbildung 8.4: *Der Roboter steht im Zentrum des Spielfelds und nutzt das spieltypische Suchverhalten um das Feld zu erkunden. Die obere linke Abbildung zeigt das Rauschen aller Perzepte über die gesamt Zeit des Experiments. Die rechte Abbildung zeigt einen Ausschnitt der Partikelfilter zu Ende des Experiments.*

stellt. Das gebildete Tormodell ist durch den schwarzen Balken illustriert. Mit Hilfe des Modells lassen sich die einzelnen Multi-Hypothesen den entsprechenden Torpfosten zuordnen. Diese Zuordnung wird durch die Farbe der Partikel visualisiert, dabei steht rot für den linken und blau für den rechten Torpfosten. Der Roboter verliert die Torpfosten während der Kopfbewegung regelmäßig. Insbesondere wird systematisch nur einer der Torpfosten wahrgenommen, sodass die korrekte Zuordnung zu den entsprechenden Multi-Hypothesen der Torpfosten an dieser Stelle interessant ist. Diese Zuordnung wird in Abbildung 8.5 (unten) visualisiert. Dabei zeigen die Farben die Zuordnung der Perzepte zu den Torpfosten (rot - links, blau - rechts). Es ist gut zu erkennen, dass bei diesem einfachen Aufbau die Zuordnung ohne einen einzigen Fehler funktioniert.

Die oberen drei Graphen der Abbildung 8.5 visualisieren den Erwartungswert sowie die Varianz der Multi-Hypothese des linken Torpfostens (blau) zur gesamten Laufzeit des Experiments. Zum Vergleich sind die entsprechenden Torpfostenperzepte (grün) ebenfalls abgetragen. Anzumerken ist, dass nur die Perzepte Beachtung finden, die im MHTM dem linken Torpfosten zugeordnet worden sind.

Gut zu beobachten ist die systematische Änderung der Distanz und des Winkels der Perzepte in Abhängigkeit der Bewegung der Kamera. Diese Veränderung kann direkt auf die unterschiedliche Position des Perzepts im Bild zurückgeführt werden (vgl. 6.1).

Darüber hinaus ist zu erwarten, dass durch eine schnelle Kopfbewegung zusätzliche Fehler eingehen.

Die Varianz, die in Abbildung 8.5 (unten) abgetragene ist, verhält sich ähnlich der Varianz aus Experiment 8.2.

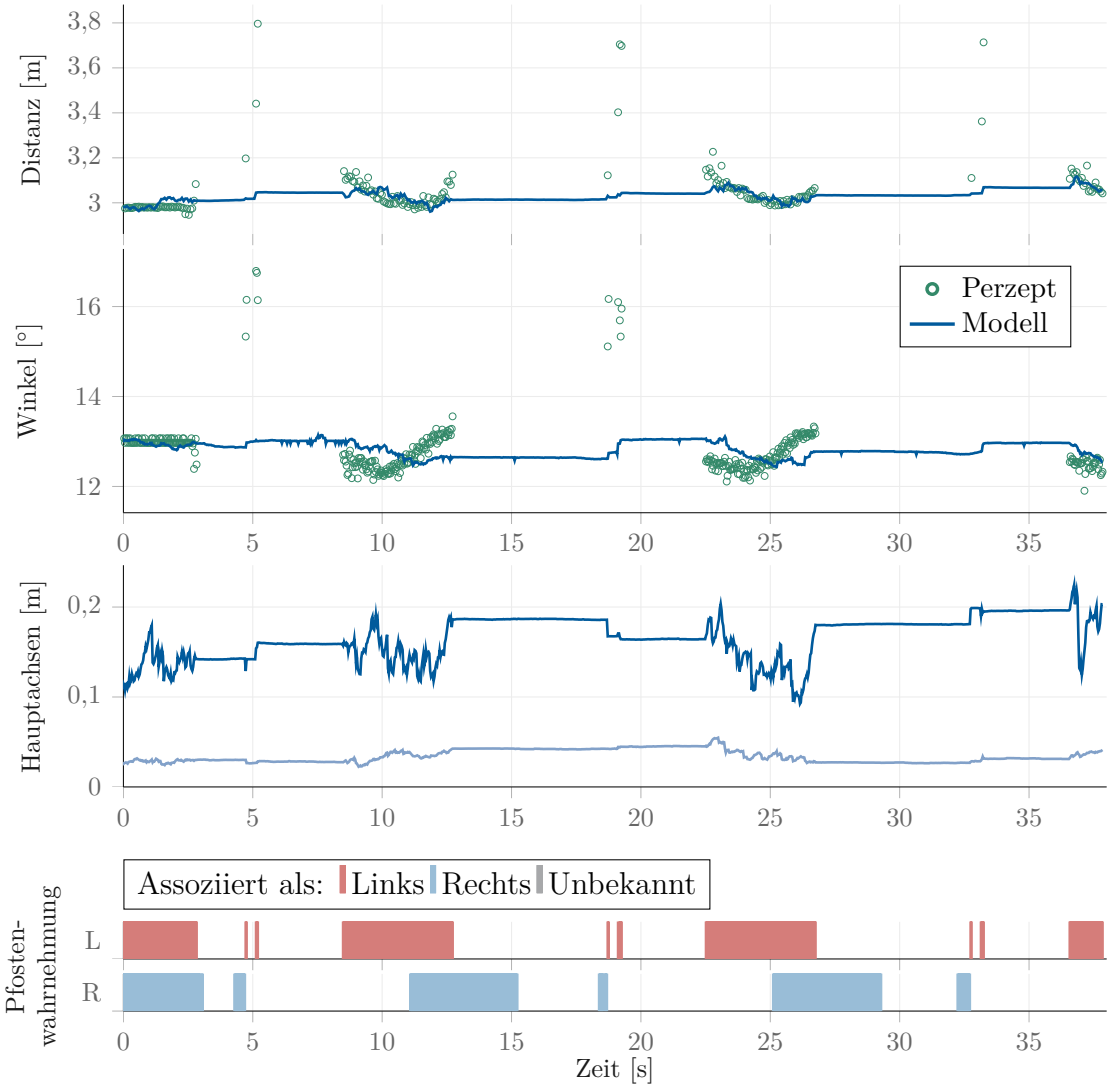

Abbildung 8.5: *Die beiden Abbildungen zeigen die Perzepte die mit dem linken Pfosten des Modells assoziiert werden. Die obere Abbildung vergleicht die Distanzen dieser Perzepte und dem Modell, die untere den entsprechenden Winkel.*

Insgesamt kann in diesem Experiment gesehen werde, wie das MHTM mit unterbrochenen Wahrnehmungen umgeht. Die einzelnen Multi-Hypothesen können sogenannte Ausreißer kompensieren. Obwohl in manchen Sequenzen nur ein Torpfosten zu sehen ist, funktioniert die Zuordnung der Perzepte zu den entsprechenden Multi-Hypothesen in diesem Fall fehlerfrei.

8.3 Experiment III: Analyse bei spärlicher Falschwahrnehmungen

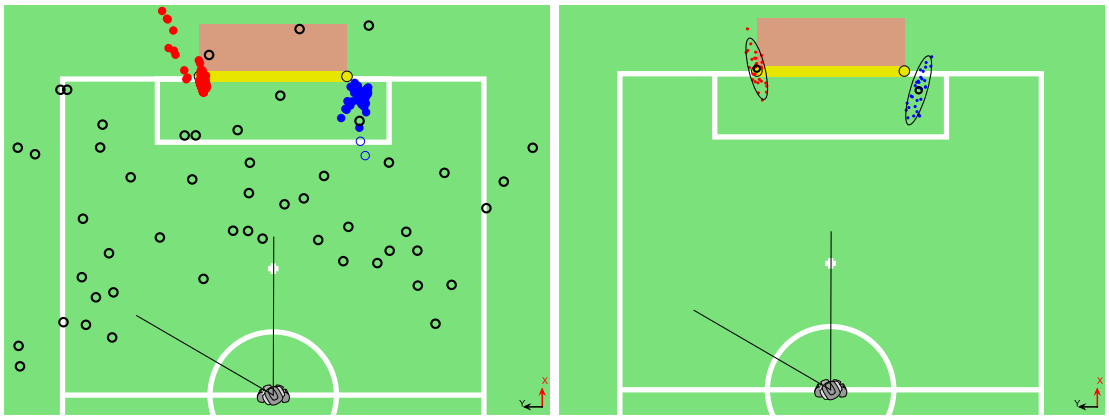

Abbildung 8.6: *Der Roboter steht still im Zentrum des Spielfeldes und nutzt die spieltypische Suchbewegung des Kopfes um das Feld zu erkunden. Dabei werden zusätzlich zufällige Wahrnehmungen eingestreut. Die linke Abbildung zeigt alle Perzepte über die gesamt Zeit des Experiments. Die Kreise markieren die wahrgenommenen Torperzepte, dabei visualisiert die Farbe die Zuordnung der Perzepte zu dem jeweiligen Pfosten des Modells: rot - links, blau - rechts, schwarz - keine (Perzeptpuffer). Die nicht ausgefüllten Kreise markieren die zusätzlich eingestreuten falschen Perzepte. Die rechte Abbildung zeigt eine Momentaufnahme der Multi-Hypothesen (Partikelfilter) am Ende des Experiments.*

In diesem Experiment wird die Stabilität des MHTMs gegenüber sporadischer Falschmessungen (engl.: *false-positives*) untersucht. Dazu werden die identischen Rohdaten aus dem Experiment 8.2 verwendet. Diese werden jedoch künstlich mit zufälligen Falschmessungen versehen. Pro Bild (Frame) wird mit einer Wahrscheinlichkeit ρ eine Falschmessung generiert. Die Messung selbst wird gleichverteilt in einem relevanten Bildabschnitt, d.h., in dem Bereich wo üblicherweise Torpfosten gesehen werden können, generiert. Mit Hilfe der Kameramatrix werden diese Perzepte in die relativen Roboterkoordinaten projiziert. Dieses zufällige Perzept wird den eigentlichen Messungen hinzugefügt und regulär vom MHTM verarbeitet. Darüber hinaus wird die Wahrscheinlichkeit der Falschmessungen ρ variiert, es wird jedoch immer maximal eine Falschmessung pro Zeitschritt erzeugt.

Die Abbildung 8.6 zeigt die gesammelten Perzepte über die gesamte Laufzeit des Experiments. Die Abbildung 8.7 veranschaulicht exemplarisch die Zuordnung der Perzepte durch das MHTM bei $\rho = 0.1$. Gut zu sehen ist, dass nahezu alle Perzepte korrekt klassifiziert werden. Zur Sekunde 24,9 kann ein unbekanntes Perzept ausgemacht werden, das fälschlicherweise als rechter Torpfosten klassifiziert wurde. Es

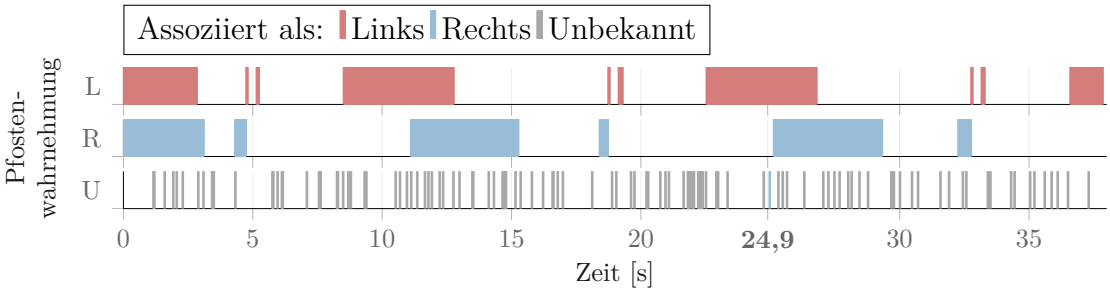

Abbildung 8.7: *Zuordnung gesehener Torpfosten. Die gesehenen Torpfosten, sowie die künstlich eingestreute Perzepte, sind als Balken in den entsprechenden drei Graphen abgetragen. Die Farbe veranschaulicht die Zuordnung dieser Perzepte mit dem MHTM. Dabei steht rot für den linken, blau für den rechten und grau für einen unbekannten Torpfosten. Bis auf vereinzelte Fälle sind alle Perzepte korrekt zugeordnet. Um die Sekunde 25 sieht man eine Falschwahrnehmung, die als linker Torpfosten identifiziert wurde.*

Wahrscheinlichkeit einer Falschwahrnehmung ρ	0.1	0.3	0.5	0.7	1.0
Mittlere Anzahl an Falschassoziationen (je 5 Experimente)	1.4 ± 1.3	5.2 ± 2.0	7.6 ± 1.5	11.2 ± 4.5	46.4 ± 39.4

Tabelle 8.2: *Anzahl falscher Assoziationen von Torperzepten in Abhängigkeit von verschiedenen Rauschstufen. Die obere Zeile gibt dabei an, mit welcher Wahrscheinlichkeit ein Zufallsperzept eingestreut wurde. Die untere Zeile ist die Anzahl falscher Assoziationen von Perzepten mit den Multi-Hypothesen gemittelt über 5 Durchläufe mit entsprechender Varianz.*

muss hier angemerkt werden, dass das Zufallsperzept unter Umständen exakt einer Multi-Hypothese sehr gut entsprochen hat. Daher deutet eine Falschklassifikation im Rahmen dieses Experiments nicht zwingend auf einen tatsächlichen Fehler hin.

Um den Einfluss von Störungen auf das MHTM systematisch zu untersuchen, wurde das Experiment mit verschiedenen Rauschstufen ρ jeweils fünf mal durchgeführt. Die Tabelle 8.2 fasst die Ergebnisse der Experimente zusammen. Bei jedem der Experimente wurde das Tormodell korrekt extrahiert. Bei drei der Durchläufe mit $\rho = 1$ wurde eine weitere (falsche) Multi-Hypothese gebildet, die aber die Bildung des korrekten Models nicht beeinflusst hat. Pro Experiment wurden 1072 tatsächliche Torperzepte in 1285 Frames verarbeitet. Die Anzahl der Falschwahr-

nehmungen nimmt insbesondere bei $\rho = 1$ stark zu, entspricht aber trotzdem nur etwa 2% aller verarbeiteten Perzepte.

Zusammenfassend lässt sich sagen, dass das MHTM dem gewählten Rauschmodell für Falschwahrnehmungen sehr gut stand halten kann.

8.4 Experiment IV: Analyse dichter Falschwahrnehmungen

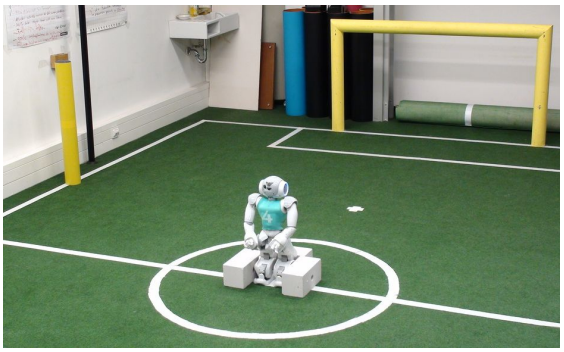

Abbildung 8.8: *Experimentaufbau mit einem zusätzlichen Torpfosten. Links im Bild sieht man einen dritten gelben Torpfosten, der aus der Sicht der Bildverarbeitung identisch mit einem regulären Torpfosten ist.*

In diesem Experiment wird das Verhalten des MHTM bei dichten Falschmessungen untersucht. Dazu wird das Experiment ähnlich zu 8.2 und 8.3 aufgebaut, d.h., der Roboter wird im Zentrum des Felds positioniert und führt mit seinem Kopf eine Suchbewegung aus. Zusätzlich wird noch ein weiterer Torpfosten auf dem Feld positioniert, der normalerweise nicht vorhanden ist. Im Kapitel 5 wurde dargelegt, dass dichte Falschmessungen aufgrund der ungeregelten Umgebung des Spielfelds möglich sind. Die Abbildung 8.8 veranschaulicht die Experimentalaufstellung.

In der Abbildung 8.9 sehen wir links die gesammelten Torperzepte über die gesamte Laufzeit des Experiments und rechts eine Momentaufnahme der Multi-Hypothesen einzelner Torpfosten am Ende des Experiments. Die Farben veranschaulichen die Zuordnung gemäß des MHTMs. Der linke Torpfosten ist rot, der Rechte blau und ein unbekanntes Perzept ist graumarkiert. Der Graph in Abbildung 8.10 visualisiert die Zuordnung einzelner Perzepte über die Zeit. Gut zu sehen ist die periodische Wahrnehmung unterschiedlicher Pfosten während der Suchbewegung. Darüber hinaus ist zu sehen, wie die Perzepte des dritten *unbekannten* Torpfostens in einer separaten Multi-Hypothese (schwarz) zusammengefasst werden und das korrekte Tor-Model extrahiert wird.

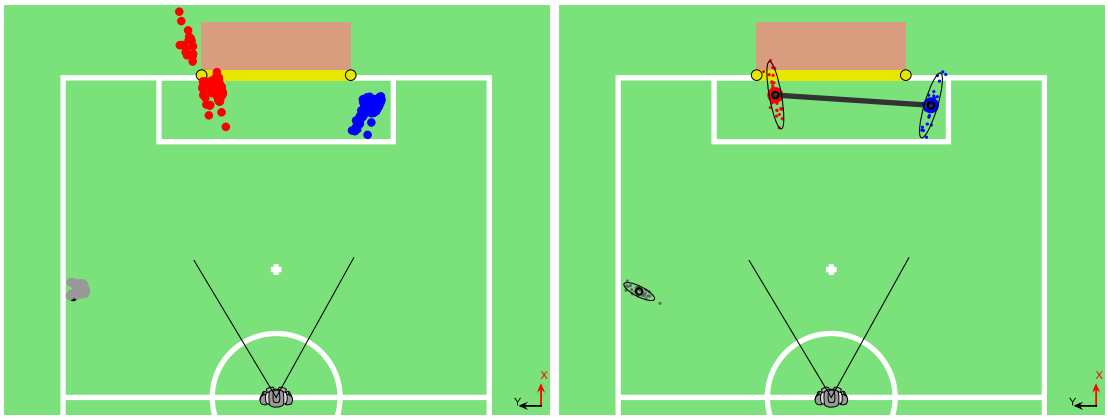

Abbildung 8.9: *Der Roboter steht still im Zentrum des Spielfeldes und nutzt die spieltypische Suchbewegung des Kopfes um das Feld zu erkunden. Ein zusätzlicher Torpfosten ist im Suchbereich des Roboters platziert. Die Abbildung auf der linken Seite veranschaulicht die Torperzepte gesammelt über den gesamten Verlauf des Experiments. Die unterschiedlichen Farben illustrieren die Zuordnung einzelner Perzepte zu den Multi-Hypothesen. Dabei steht rot für den linken, blau für den rechten und grau für einen unbekannten Torpfosten. Das Bild rechts zeigt die Momentaufnahme einzelner Multi-Hypothesen am Ende des Experiments.*

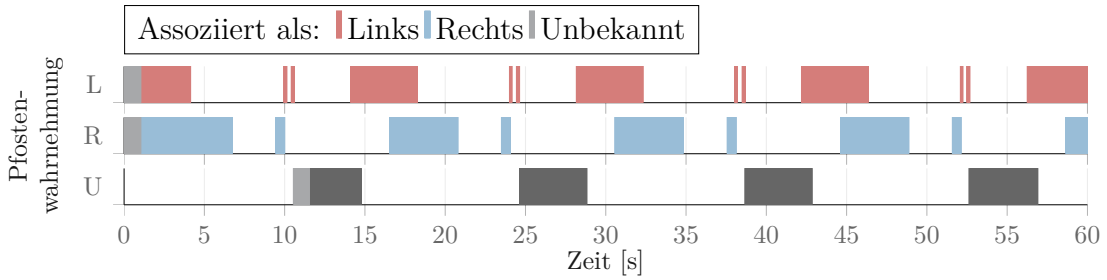

Abbildung 8.10: *Torperzepte zugeordnet gemäß des MHTMs: rot - links, blau - rechts, dunkelgrau - unbekannt, hellgrau Perzeptpuffer.*

8.5 Experiment V: Laufen zum Tor

In diesem Experiment wird das Verhalten des HMTMs während der Bewegung des Roboters untersucht. Dazu startet der Roboter im eigenen Tor und läuft über das gesamte Feld gerade in das gegnerische Tor. Seine Position wird während dessen vom Tracking-System erfasst, so dass genaue Positionsdaten des Roboters zum Vergleich zur Verfügung stehen. Die Abbildung 8.11 veranschaulicht den Experimentalaufbau.

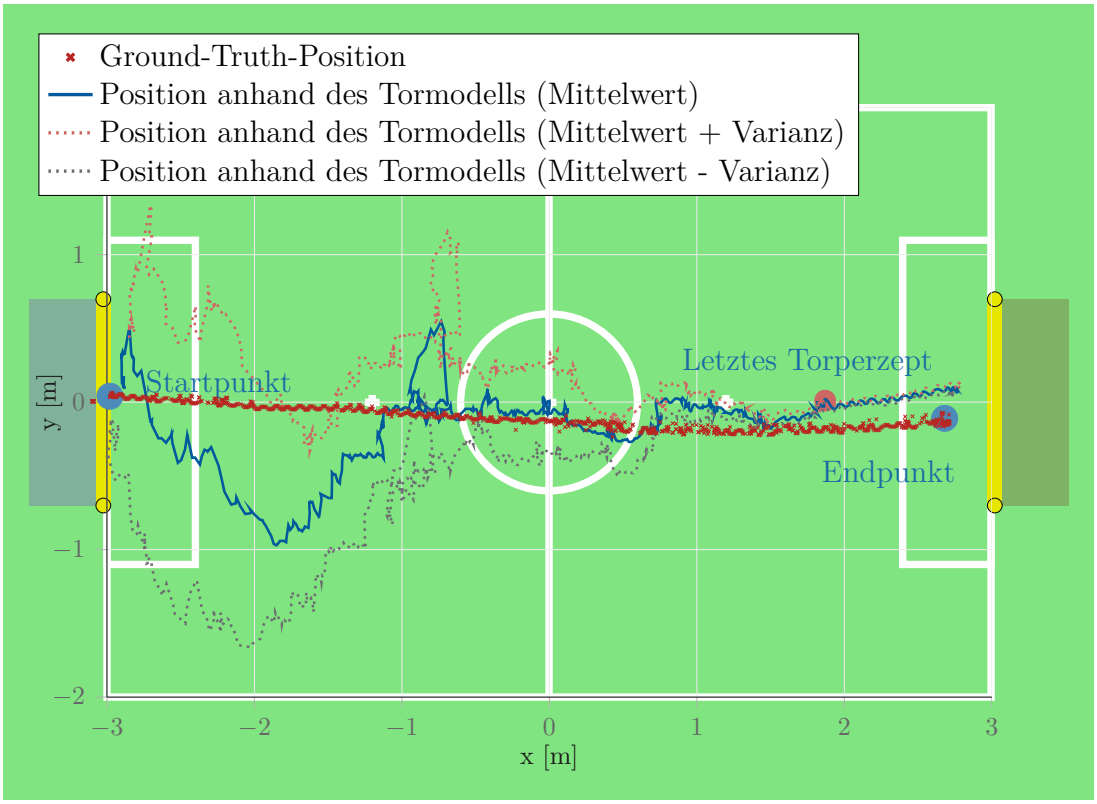

Abbildung 8.11: *Der Roboter startet im eigenen Tor und läuft zum Ball, der sich im gegnerischen Tor befindet. Der rote Graph zeigt den zurückgelegten Weg des Roboters gemessen durch das Tracking-System. Start und End-Punkte sind entsprechend markiert. Der blaue Graph zeigt die Position des Roboters berechnet anhand des Tormodels und gemittelt über 10 Durchläufe. Die punktierten Linien illustrieren die Varianz der Roboterposition über die 10 Experimente. Der Punkt der letzten Wahrnehmung eines Torpfostens ist ebenfalls eingetragen.*

In Abbildung 8.11 ist der vom Roboter zurückgelegte Weg veranschaulicht. Die Graphen zeigen zum einem die tatsächliche Position des Roboters und zum ande-

rem die anhand des Tormodells rekonstruierte Position des Roboters. Die Letztere ist über zehn verschiedene Durchläufe gemittelt. Als Eingabe dienten die identische Sequenz von Perzepten. Der blaue Graph in Abbildung 8.11 visualisiert die gemittelte Position zusammen mit der Varianz (punktierte Graphen). Beim Berechnen der Roboterposition anhand des Tormodells wird angenommen, dass dieses Tormodell dem eigentlichen Tor entspricht. Auf diese Weise spiegeln sich die Fehler in der Ausrichtung (Rotation) des Tormodels als Translation auf einem Kreis um das Tor wieder. Das führt zu den erwarteten starken Schwankungen der berechneten Roboterposition an der y-Axis. Diese Schwankungen nehmen mit kürzerer Distanz zum Tor deutlich ab. Auch das Rauschen des Modells nimmt mit kleiner werdender Distanz erwartungsgemäß ab. Gut zu sehen ist ebenfalls die Stelle ab der, aufgrund zu kleiner Distanz, kein Torpfosten mehr gesehen werden kann (dieser Punkt ist gesondert in der Abbildung 8.11 markiert). Ab dieser Stelle wurde das Modell nur noch anhand der Bewegungsdaten fortgesetzt.

In der Abbildung 8.12 (unten) wird die Zuordnung der Perzepte zu den Torpfosten veranschaulicht. Am Anfang des Experiments sieht man gut die Phase, in der das Modell aufgebaut wird. In dieser Phase werden alle Perzepte als unbekannt markiert. Gut zu sehen ist auch, dass die Wahrnehmung der Perzepte unterbrochen ist. Dies resultiert aus dem Verhalten des Roboters. Während der Annäherung zum Ball schaut er sich stets um. Trotz der Unterbrechung und der Bewegung des Roboters gibt es keine Fehlzuordnungen.

Die Abbildung 8.12 veranschaulicht in den oberen zwei Graphen die Distanz und den Winkel der Multi-Hypothese des linken Torpfostens mit den entsprechend zugeordneten Perzepten. Man kann sehen, dass der Torpfosten des MHTMs den Perzepten folgt. Die Abweichung in der Distanz im Vergleich zur Ground Truth liegt maximal im Bereich von etwa 2 m, nimmt aber mit kürzerer Entfernung rapide ab. Der dritte Graph von oben zeigt die Varianz dieser Multi-Hypothese entlang der Hauptachsen. Es lässt sich sehr gut erkennen, dass die Varianz abnimmt, je näher der Roboter dem gegnerischen Tor kommt. Auch die Asymmetrie der Verteilung nimmt stark ab, d.h., die Hauptachsen sind ähnlich lang, da der Einfluss des Distanzrauschens mit Verringerung der Entfernung stark abnimmt und das Modell so deutlich genauer werden kann.

Dieses Experiment zeigt insbesondere, dass das MHTM in der Lage ist, das Tor in einer realitätsnahen Situation reproduzierbar und robust zu verfolgen.

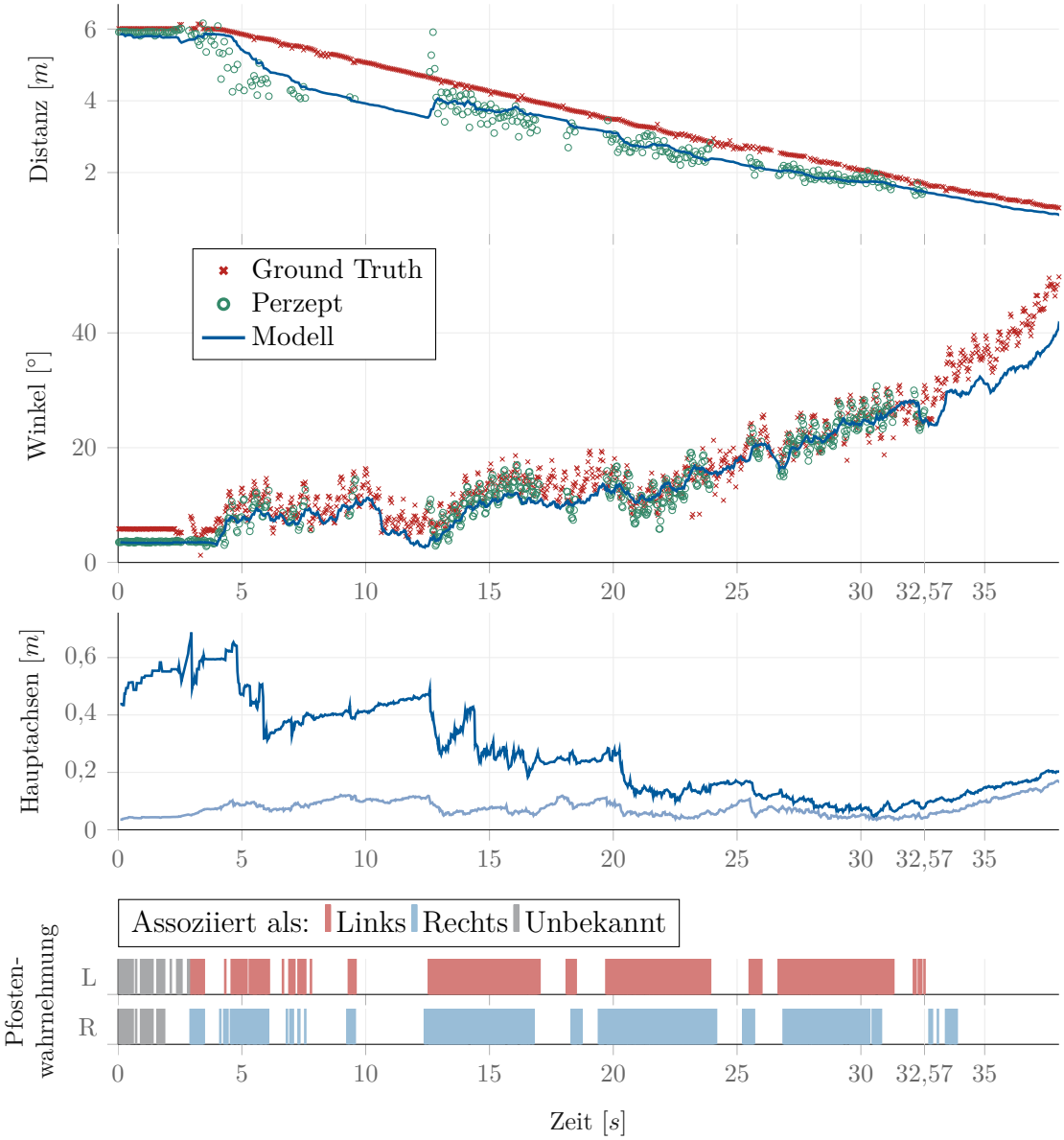

Abbildung 8.12: *Die oberen zwei Graphen zeigen die Distanz und den Winkel zum linken Torpfosten. Eingezeichnet sind Ground Truth gemäß des Trackingsystems (rot), Torperzepte (grün) und das Modell des Torpfostens (blau). Der dritte Graph von oben illustriert die Hauptachsen der Partikelverteilung der Multi-Hypothese des Modells für den linken Torpfosten (dunkel - erste Hauptkomponente, hell - zweite Hauptkomponente). Der untere Graph visualisiert die Perzepte der Torpfosten über die Zeit, dabei illustrieren die Farben deren Zuordnung gemäß MHTM (rot - links, blau - rechts, grau - unbekannt).*

8.6 Zusammenfassung

Die Experimente untersuchen die Funktionsweise einzelner Stufen des MHTM. Insbesondere ist zu erkennen, dass die einzelnen Multi-Hypothesen, basierend auf Partikelfiltern, eine gute Filter- und Tracking-Leistung zeigen. Die Propagierung semantischer Information des Tormodels zur Klassifikation einzelner Perzepte und deren Zuordnung zu den Torpfosten zeigen sich sehr robust und wirken sehr vielversprechend. Es konnte gezeigt werden, dass das MHTM robust gegenüber sporadischen Falschwahrnehmungen ist, welche durch den temporalen Perzeptpuffer ausgefiltert werden. Die dichten Falschwahrnehmungen münden in einer separaten Multi-Hypothese und können auf semantischer Ebene bei der Extraktion des Modells herausgefiltert werden.

Der integrierte Experiment 8.5 hat gezeigt, dass das MHTM unter realitätsnahen Bedingungen das Tor zuverlässig verfolgen kann.

Es ist sicherlich eine Reihe weiterer Experimente notwendig um die Robustheit und die Grenzen des MHTM unter realistischeren Bedingungen zu testen. Insbesondere ist es von Interesse, wie sich der Filter bei schlechten Vorhersagen entwickelt. Wie im Experiment 7.6 gezeigt, können in Spielsituationen drastische Odometrfiefehler entstehen. Daraus resultieren schlechte Vorhersagen des Partikelfilters. Von Interesse ist, wie sich diese Situationen auf ein gesamtes Spiel auswirken und wie oft der Roboter dennoch zuverlässig ein Tormodell bereitstellen kann. Die vorgestellte Ergebnisse deuten darauf hin, dass der gewählte Ansatz sehr vielversprechend ist.

9 Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde ein Modell eines Fußballtors im RoboCup entwickelt und implementiert. Entsprechend der Zielstellung wurde dazu die Idee der Bayes'schen Modellierung vorgestellt und verwendet. Das Modell stellt dem Roboter die relative Position zweier Torpfosten des Fußballtors zur Verfügung. Darüber hinaus modelliert es dichte Inkonsistenzen der Umwelt explizit mit. Diese können dann auf der semantischen Ebene interpretiert werden. Spärliche Inkonsistenzen werden durch Assoziation mit Multi-Hypothesen und einem Puffer abgefangen und haben nur geringen Einfluss auf das Modell.

In einer Reihe von Experimenten wurde die Güte der zugrunde liegenden Sensoreingaben im Kapitel 6 und 7, die grundlegenden Funktionsweisen der Implementation im Kapitel 8 empirisch überprüft.

Dieses Kapitel fasst die Erkenntnisse der drei Kapitel zusammen. Darüber hinaus wird in jedem Abschnitt ein Ausblick herausgearbeitet, der die zukünftigen Untersuchungen beschreibt.

9.1 Wahrnehmungsanalyse

Um die Methoden zur Realisierung eines Tormodells adäquat wählen zu können, wurde eine Übersicht von Fehlerquellen der Objekterkennung und Bewegungsmodellierung (Odometrie) des zugrunde gelegten NaoTH-Frameworks empirisch ermittelt. Dabei wurden die Untersuchungen so gestaltet, dass sie ebenfalls eine Grundlage für zukünftige Modellierungsaufgaben bilden.

Es wurde gezeigt, dass die Fehler aus der Objekterkennung (bezüglich der Objektposition im Bild) einer Regelmäßigkeit folgen (vgl. 6.1). Aufgrund der fließend ansteigenden Entfernungsmessung liegt die Vermutung nahe, dass die Fehler durch die ungenauen intrinsischen Kameraparameter verursacht werden. Darüber hinaus sind vor allem im Randbereich eines Bildes sehr hohe, sprunghafte Falschmessungen zu erkennen. Das Experiment 8.2 konnte dies aufzeigen. Die zugrunde liegenden Fehler sind für verschiedene Roboter nicht einheitlich. Ein gemeinsames, roboterunabhängiges Sensormodell kann aus ihnen nicht hergeleitet werden. Wenn man die Fehler reduzieren möchte ist es notwendig, die intrinsischen Parameter der Kamera exakt zu vermessen und sie bei der Ermittlung der Kameramatrix zu berücksichtigen (vgl. 2.3.3).

Im Experiment 6.2 wurde ein systematischer Fehler entdeckt. Die Experimente zeigen bei vier Robotern einen sehr ähnlichen Verlauf. Je nach horizontaler Kopfposition und bei gleichzeitiger konstanter Entfernung des Objekts, wurden Messungsungsungenauigkeiten von über einem Meter festgestellt. Eine Lookup-Tabelle oder die Überprüfung des Algorithmus können hier zu einer deutlich verbesserten Wahrnehmung führen. Wenngleich der Roboter im aktuellen Verhalten seinen Kopf nicht über einen Winkel von 40° bzw. −40° auslenkt, so sollte man in zukünftigen Arbeiten die Systematik ausnutzen und den Fehler auf wenige Zentimeter einschränken.

Auch die empirische Analyse der Odometrieberechnung ist aufschlussreich und bildet eine sehr gute Grundlage für zukünftige Arbeiten. Es wurde gezeigt, dass sich vier unterschiedliche Roboter bei konstanter Geradeausbewegung stets langsamer einschätzen. Dies zeigen vor allem die Experimente 7.1 und 7.3.

Darüber hinaus kann man den Experimenten entnehmen, dass die Bewegung senkrecht zur Gehrichtung für jeden Roboter unterschiedlich ist. Die Tendenz ist teilweise entgegengesetzt.

Des Weiteren wurde die Drehung um die eigene Achse im Experiment 7.2 analysiert. Erneut wurden vier unterschiedliche Roboter eingesetzt. Diese Roboter zeigen für ihre Drehung nach links bzw. nach rechts ein unterschiedliches Verhalten. Darüber hinaus scheint es so, als können die Roboter eine Rechtsdrehung stabiler und genauer durchführen. Es ist bekannt, dass der Roboter nicht exakt symmetrisch gebaut ist. Daraus kann ein systematischer Fehler resultieren. Bezüglich der Linksdrehung ist zu erkennen, dass die Roboter diese unterschiedlich schnell absolvieren. Darüber hinaus sind am Ende von drei Drehungen Unterschiede im Fehler von über 200° erkennbar. Dies sind beträchtliche Unterschiede.

Zusammenfassend lässt sich sagen, dass die lokalen Unterschiede in der Bildverarbeitung gering sind. Darüber hinaus sind die Fehler der Odometrieberechnung für kurze Zeitabschnitte ebenfalls sehr gering. Der Ansatz des lokalen Multi-Hypothesen-Tormodells, anhand der Odometrie die Vorhersage des Zustandes zu treffen, ist dadurch durchaus legitim. Dass die Fehler der Sensoreingaben offensichtlich nicht normalverteilt sind, stützt die Entscheidung für den Partikelfilter.

Zukünftige Untersuchungen müssen sich in verschiedene Richtungen orientieren. Einerseits gilt es die Ursache der systematischen Fehler genau einzugrenzen und zu beheben. Andererseits sind statistische Analysen in Bezug auf die Odometriefehler hilfreich. Die Odometrie wird in unseren Ansätzen zur Vorhersage des Modells verwendet. Fehler in der Odometrie können zu beträchtlichen Fehlern in einem weit entfernten Modell führen. Mithilfe einer geeigneten Analyse ließe sich bestimmen, unter welchen Gegebenheiten eine Vorhersage unsicherer wird. Sehr wahrscheinlich ist davon auszugehen, dass man einem Modell bezüglich eines weit entfernten Objekts kürzer vertraut, als einem Modell bezüglich eines sehr nahen Objekts. Die

Schwierigkeit einer exakten Untersuchung ist aus den Spielbeobachtungen ableitbar. Roboter verändern ihren Gang während eines Spiels. Dies liegt vor allem in der Wärmeentwicklung in den Gelenken begründet. Eine Veränderung der Messgenauigkeit ist die Folge. Darüber hinaus verschleißen Zahnräder und Gelenke der Roboter über längere Zeit. Dennoch kann durch eine Analyse zumindest eine Heuristik für das Bewegungsmodell abgeleitet, und so die Modellierung verbessert werden.

Es ist weiterhin denkbar, die Parameter des Bewegungs- und Sensormodells lernen zu lassen. Das Einstellen von Parametern ist aufgrund des Verschleißes und der Wärmeentwicklung mühsam. Ein Roboter sollte während eines Spieles diese Parameter daher schrittweise verbessern. So ist es zum Beispiel möglich, die Vorhersage mit den tatsächlichen Sichtungen abzugleichen. Einer konstanten Falschvorhersage, bspw. bedingt durch einen Sturz, könnte so entgegengewirkt werden.

Ein erster einfacher Ansatz wäre es, dass das Modell zwischen Gehen und Stehen unterschiedet. In vielen Situationen stehen die Roboter auf dem Spielfeld. Eine Vorhersage über die zukünftige Position des lokalen Tormodells ist für einen stehenden Roboter nicht notwendig. Das Rauschen des Bewegungsmodells kann so deutlich besser auf eine tatsächliche Bewegung angewendet werden, und muss nicht den Sonderfall eines stehenden Roboters mit abdecken.

9.2 Muti-Hypothesen-Tormodell

In dieser Arbeit wurde ein Multi-Hypothesen-Tormodell vorgestellt und seine grundlegende Funktionsweise analysiert. Sehr vielversprechend wirkt die Anwendung bezüglich des Umgangs mit Inkonsistenzen. Hier wird zwischen dichten und spärlichen Falschmessungen unterschieden. Diese werden auf verschiedenen Ebenen des MHTMs behandelt. Dazu wurden im Kapitel 8 einige Experimente durchgeführt. Es wurde gezeigt, dass das Multi-Hypothesen-Tormodell mit dem Mechanismus des Perzeptpuffers spärliche Falschmessungen ignoriert. In vielen Implementationen haben Falschmessungen direkten Einfluss auf einen Filter und verrauschen diesen. Das vorgestellte Multi-Hypothesen-Tormodell erzeugt hingegen ein sehr stabiles Modell (vlg. 8.3). Voraussetzung hierfür ist die Annahme, dass ein Perzept genau einen Pfosten darstellt. Für Tracking-Algorithmen, in denen mehrere Objekte in einem Blob zusammengefasst werden, ist diese Annahme nicht gültig und der Ansatz nicht umsetzbar. Für Ansätze, die Objekte charakterisieren und unabhängig tracken, kann diese Methode zur Verbesserung der Trackingleistung angewandt werden.

Die Natur des Filters erlaubt es weiterhin, eine dichte Falschmessung in einer eigenen Multi-Hypothese zu pflegen und Pfostenperzepte dieser Multi-Hypothese zuzuordnen (vlg. 8.4). Dichte (dauerhafte) inkonsistente Pfostensichtungen kön-

nen so gepflegt werden. In den meisten Implementationen eines Lokaliserungsalgorithmus für Roboter wird die Markow-Annahme ausgenutzt. Diese besagt, dass alle vorhergehenden Sichtungen und Zustände in dem aktuellen Zustand enthalten sind. Insbesondere bedeutet dies, dass auch inkonsistente Sichtungen enthalten sind und den Zustand beeinflussen. Dies gilt für *einfache* Partikelfilter genauso wie für Multi-Hypothesen-Ansätze, die für einen neuen Torpfosten eine neue Hypothese erstellen und tracken. Anhand der expliziten Modellierung von Inkonsistenzen kann vermieden werden, dass ein solcher Torpfosten in die Berechnung des Lokalisierungsfilters einfließt. Dies kann zu einer stabileren Zustandsbeschreibung führen.

Dazu ist es allerdings notwendig, dass man beide tatsächlichen Pfosten gesehen hat. Nur durch ein zuverlässiges Modell kann eine inkonsistente Sichtung ausgeschlossen werden. Der wahrscheinlich positive Einfluss, den das Multi-Hypothesen-Tormodell auf die Lokalisierung hat, muss allerdings noch experimentell bestimmt werden. Eine Möglichkeit wäre der Vergleich der Unsicherheiten unterschiedlicher Implementationen. Die Roboterposition könnte unter Anwendung des MHTMs ermittelt werden. Diese Position könnte der aktuellen Implementation entgegengestellt werden.

Bei einem positiven Ergebnis einer solchen Analyse, könnte man den Ansatz daraufhin ebenso zur Modellierung der Linien verwenden. So könnte man das Modell für eine größere Anzahl von mehrdeutigen Perzepten erproben.

Das Multi-Hypothesen-Tormodell kann außerdem auch auf seine Parameter untersucht werden. Die Anzahl der verwendeten Partikel je Filter sind ein Kriterium für die aufgewendete Rechenleistung des Filters (vgl. 3.4). Untersuchungen zur optimalen Anzahl der zu verwendenden Partikel sind daher notwendig. Darüber hinaus könnte die Anzahl mit zunehmender Unsicherheit variieren. Denkbar ist auch hier, dass der Roboter die passende Anzahl erlernt.

Bisher werden Hypothesen über eine bestimmte Zeit gepflegt. Wenn sie lange keine Assoziation mehr erfahren haben, werden sie gelöscht. Dieser Paramter ist fest und könnte an verschiedene Umstände gekoppelt werden.

Darüber hinaus fließen keine semantischen Informationen in die Entscheidung zur Löschung einer Multi-Hypothese ein. Es ist denkbar, dass durch ein hohes Rauschen der Perzepte eine Multi-Hypothese, die einen linken Pfosten repräsentiert, im nächsten Zeitschritt einen rechten Torpfosten zugeordnet wird. Dieser Sprung wird derzeit nicht behandelt und könnte die Unsicherheit der Multi-Hypothesen vergrößern.

Die Extraktion des Modells sollte darauf Einfluss nehmen. Die passenden Constraintverfahren können dabei umfassend in einer zukünftigen Arbeit untersucht werden.

Literaturverzeichnis

[AH14] Anderson, Peter ; Hengst, Bernhard: Fast Monocular Visual Compass for a Computationally Limited Robot. In: Behnke, Sven (Hrsg.) ; Veloso, Manuela (Hrsg.) ; Visser, Arnoud (Hrsg.) ; Xiong, Rong (Hrsg.): *RoboCup 2013: Robot World Cup XVII* Bd. 8371. Springer Berlin Heidelberg, 2014. – ISBN 978–3–662–44467–2, S. 244–255

[Ald13] Aldebaran Robotics: *Dokumentation des Naos von Aldebaran Robotics*. Version: 2013. `http://community.aldebaran.com/doc`, Abruf: 11. November 2013

[BBM09] Bazzani, Loris ; Bloisi, Domenico ; Murino, Vittorio: A comparison of multi hypothesis kalman filter and particle filter for multi-target tracking. In: *Performance Evaluation of Tracking and Surveillance workshop at IEEE Computer Society Conference on CVPR*. Miami, Florida, 2009, S. 47–54

[BGH+13] Barrett, Samuel ; Genter, Katie ; He, Yuchen ; Hester, Todd ; Khandelwal, Piyush ; Menashe, Jacob ; Stone, Peter: The 2012 UT Austin Villa Code Release. In: *RoboCup-2013: Robot Soccer World Cup XVII*, 2013

[BH10] Burkhard, Hans-Dieter ; Hafner, Verena: *Vorlesung Kognitive Robotik*. 2010. – Halbkurs der Humboldt-Universität zu Berlin

[BST75] Bar-Shalom, Yaakov ; Tse, Edison: Tracking in a Cluttered Environment with Probabilistic Data Association. In: *Automatica* 11 (1975), Nr. 5, S. 451–460

[CKU+10] Czarnetzki, Stefan ; Kerner, Sören ; Urbann, Oliver ; Hofmann, Matthias ; Stumm, Sven ; Schwarz, Ingmar: Nao Devils Dortmund Team Report 2010 / Robotics Research Institute, TU Dortmund University. 2010. – Forschungsbericht

[Cox93] Cox, Ingemar J.: A Review of Statistical Data Association Techniques for Motion Correspondence. In: *International Journal of Computer Vision* 10 (1993), Nr. 1, S. 53–66

[EGZ+04] Elatta, AY ; Gen, Li P. ; Zhi, Fan L. ; Daoyuan, Yu ; Fei, Luo: An overview of robot calibration. In: *Information Technology Journal* 3 (2004), Nr. 1, S. 74–78

[FBSS83] Fortmann, Thomas E. ; Bar-Shalom, Yaakov ; Scheffe, Molly: Sonar tracking of multiple targets using joint probabilistic data association. In: *IEEE Journal of Oceanic Engineering* 8 (1983), Juli, Nr. 3, S. 173–184

[Göh09] Göhring, Daniel: *Constraint Based World Modeling for Multi Agent Systems in Dynamic Environments*, Humboldt-Universität zu Berlin, Diss., 2009

[GS08] Guerrero, Pablo ; Solar, Javier Ruiz-del: Improving Robot Self-Localization using Landmarks' Poses Tracking and Odometry Error Estimation. In: *RoboCup 2007: Robot Soccer World Cup XI*. Springer, 2008, S. 148–158

[GSS93] Gordon, Neil J. ; Salmond, David J. ; Smith, Adrian F.: Novel approach to nonlinear/non-Gaussian Bayesian state estimation. In: *IEE Proceedings F (Radar and Signal Processing)* Bd. 140 IET, 1993, S. 107–113

[Hol13] Holzhauer, Florian: *Nao Team Humboldt: Entwicklung und Qualitätssicherung*, Humboldt-Universität zu Berlin, Diplomarbeit, 2013

[IB96] Isard, Michael ; Blake, Andrew: Contour tracking by stochastic propagation of conditional density. In: *In Proc. european Conf. Computer Vision*. Cambridge, UK : Springer, 1996, S. 343–356

[Kal60] Kalman, Rudolph E.: A New Approach to Linear Filtering and Prediction Problems. In: *Journal of basic Engineering* 82 (1960), Nr. 1, S. 35–45

[KMA01] Koller-Meier, Esther B. ; Ade, Frank: Tracking multiple objects using the Condensation algorithm. In: *Robotics and Autonomous Systems* 34 (2001), Nr. 2–3, S. 93–105. – ISSN 0921–8890. – European Workshop on Advanced Mobile Robots

[KMSB13] Kaden, Steffen ; Mellmann, Heinrich ; Scheunemann, Marcus ; Burkhard, Hans-Dieter: Voronoi Based Strategic Positioning for Robot Soccer. In: Szczuka, Marcin S. (Hrsg.) ; Czaja, Ludwik (Hrsg.) ; Kacprzak, Magdalena (Hrsg.): *Proceedings of the 22nd International Workshop on Concurrency, Specification and Programming (CS&P)*

Bd. 1032. Warsaw, Poland : CEUR-WS.org, 2013 (CEUR Workshop Proceedings), S. 271–282

[KRL14] Kastner, Tobias ; Röfer, Thomas ; Laue, Tim: Automatic robot calibration for the NAO. In: *RoboCup 2014: Robot Soccer World Cup XVIII*. Noch nicht veroffentlicht, 2014

[LBBJ04] Lötzsch, Martin ; Bach, Joscha ; Burkhard, Hans-Dieter ; Jüngel, Matthias: Designing agent behavior with the extensible agent behavior specification language XABSL. In: *RoboCup 2003: Robot Soccer World Cup VII*. Springer, 2004, S. 114–124

[MB00] MacCormick, John ; Blake, Andrew: A Probabilistic Exclusion Principle for Tracking Multiple Objects. In: *International Journal of Computer Vision* 39 (2000), Nr. 1, S. 57–71

[Mel10] Mellmann, Heinrich: *Ein anderes Modell der Welt: Alternative Methoden zur Lokalisierung Mobiler Roboter*, Humboldt Universität zu Berlin, Diplomarbeit, April 2010

[MS11] Mellmann, Heinrich ; Scheunemann, Marcus: Local Goal Model for a Humanoid Soccer Robot. In: Marcin Szczuka, Andrzej Skowron Magdalena K. Ludwik Czaja C. Ludwik Czaja (Hrsg.): *Proceedings of the Workshop on Concurrency, Specification, and Programming CS&P 2011*. Pułtusk, Poland : Białystok University of Technology, September 2011, S. 353–360

[MSBH14] Mellmann, Heinrich ; Scheunemann, Marcus ; Burkhard, Hans-Dieter ; Hafner, Verena: Berlin United – NaoTH 2014 / Kognitive Robotik, Humboldt-Universität zu Berlin. 2014. – Forschungsbericht

[MSS13] Mellmann, Heinrich ; Scheunemann, Marcus ; Stadie, Oliver: Adaptive Grasping for a Small Humanoid Robot Utilizing Force- and Electric Current Sensors. In: Szczuka, Marcin S. (Hrsg.) ; Czaja, Ludwik (Hrsg.) ; Kacprzak, Magdalena (Hrsg.): *Proceedings of the 22nd International Workshop on Concurrency, Specification and Programming (CS&P)* Bd. 1032. Warsaw, Poland : CEUR-WS.org, 2013 (CEUR Workshop Proceedings), S. 283–293

[Nao13] Nao Team Humboldt (Hrsg.): *Offizieller Internetauftritt der Arbeitsgruppe*. Version: 2013. `http://naoth.de`, Abruf: 01. Juli 2013

[OpT14] *Präsentation des Traciking-Systems OptiTrack der Firma NaturalPoint.* Version: 2014. `http://naturalpoint.com/optitrack/`, Abruf: 11. Mai 2014

[PS99] Pitt, Michael K. ; Shephard, Neil: Filtering via simulation: Auxiliary particle filters. In: *Journal of the American statistical association* 94 (1999), Nr. 446, S. 590–599

[RCD13] *Eigenbeschreibung der RoboCup-Foundation.* Version: 2013. `http://major.robocupgermanopen.de/de/robocup`, Abruf: 11. Mai 2013

[RCP11] *Offizielle Internetseite der RoboCup-Foundation.* Version: 8 2011. `http://robocup.org`, Abruf: 1. August 2011

[Rei79] Reid, Donald B.: An Algorithm for Tracking Multiple Targets. In: *IEEE Transactions on Automatic Control* 24 (1979), Nr. 6, S. 843–854

[RLM+13] Röfer, Thomas ; Laue, Tim ; Müller, Judith ; Bartsch, Michel ; Batram, Malte J. ; Böckmann, Arne ; Böschen, Martin ; Kroker, Martin ; Maass, Florian ; Münder, Thomas ; Steinbeck, Marcel ; Stolpmann, Andreas ; Taddiken, Simon ; Tsogias, Alexis ; Wenk, Felix: B-Human Team Report and Code Release 2013. 2013. – Forschungsbericht

[RLM+14] Röfer, Thomas ; Laue, Tim ; Müller, Judith ; Bartsch, Michel ; Beenenga, Jonas ; Jenett, Dana ; Kastner, Tobias ; Klose, Vanessa ; Koralewski, Sebastian ; Maass, Florian ; Maier, Elena ; Meissner, Paul ; Schüthe, Dennis ; Siemer, Caren ; Vosteen, Jan-Bernd: B-Human Team Description for RoboCup 2014. In: *RoboCup 2014: Robot Soccer World Cup XVIII Preproceedings.* João Pessoa, Brasilien : RoboCup Federation, 2014

[SBFC01a] Schulz, Dirk ; Burgard, Wolfram ; Fox, Dieter ; Cremers, Armin B.: Tracking multiple moving objects with a mobile robot. In: *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001)* Bd. 1 IEEE, 2001, S. I–371

[SBFC01b] Schulz, Dirk ; Burgard, Wolfram ; Fox, Dieter ; Cremers, Armin B.: Tracking Multiple Moving Targets with a Mobile Robot using Particle Filters and Statistical Data Association. In: *Proc. IEEE International Conference on Robotics and Automation* Bd. 2 IEEE, 2001, S. 1665–1670

[SBFC03] Schulz, Dirk ; Burgard, Wolfram ; Fox, Dieter ; Cremers, Armin B.: People tracking with mobile robots using sample-based joint probabilistic data association filters. In: *The International Journal of Robotics Research* 22 (2003), Nr. 2, S. 99–116

[Sit64] Sittler, Robert W.: An optimal data association problem in surveillance theory. In: *IEEE Transactions on Military Electronics* 8 (1964), Nr. 2, S. 125–139

[Tas14] Tasse, Stefan: *Stochastic filtering on mobile devices in complex dynamic environments*, Technische Universität Dortmund, Diss., 3 2014

[TBF06] Thrun, Sebastian ; Burgard, Wolfram ; Fox, Dieter: *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*. The MIT Press, 2006 (Intelligent robotics and autonomous agents). – ISBN 13–978 0 262 201629

[Tec11] Techical Committe SPL (Hrsg.): *Offizielle Informationsseite der Standard Platform League*. Version: 2011. `http://tzi.de/spl/`, Abruf: 01. August 2011

[VGP05] Vermaak, Jaco ; Godsill, Simon J. ; Perez, Patrick: Monte carlo filtering for multi target tracking and data association. In: *IEEE Transactions on Aerospace and Electronic Systems* 41 (2005), Nr. 1, S. 309–332

[YMC07] Yu, Qian ; Medioni, Gérard ; Cohen, Isaac: Multiple Target Tracking Using Spatio-Temporal Markov Chain Monte Carlo Data Association. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* IEEE, 2007, S. 1–8

Selbständigkeitserklärung

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und noch nicht für andere Prüfungen eingereicht habe. Sämtliche Quellen einschließlich Internetquellen, die unverändert oder abgewandelt wiedergegeben werden, insbesondere Quellen für Texte, Grafiken, Tabellen und Bilder, sind als solche kenntlich gemacht. Mir ist bekannt, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

Berlin, den 4. September 2014 .....

Danksagung

Die letzten Jahre haben meine Interessen nachhaltig positiv geprägt und meinen Fokus auf ein vielseitiges und spannendes Betätigungsfeld gerichtet. Einen ganz besonderen Dank möchte ich an dieser Stelle an Prof. Dr. Hans-Dieter Burkhard richten, der mich mit seiner Begeisterung für den RoboCup und die Robotik angesteckt hat, und so die Tür für ein aufregende Arbeit öffnete. Für die damit verbundenen Erfahrungen und Bekanntschaften bin ich genauso dankbar, wie für die vielen spannenden Gespräche.

Nicht weniger beeinflusst wurde ich durch die inspirierende Arbeit von Prof. Verena Hafner. Durch ihre Vorlesungen und ihre Forschung zur *Embodied AI* wurde mein Blickwinkel um weitere spannende Aspekte erweitert. Die Versuche, diese Gedanken mit in das RoboCup-Team einfließen zu lassen, wurden durch stetig bessere Platzierungen im Wettkampf honoriert. Liebe Frau Hafner, lieber Herr Burkhard: Vielen Dank für das viele übermittelte Wissen, Ihr Vertrauen und die zeitliche sowie finanzielle Investition in meine Person.

In den letzten vier Jahren war ich Mitglied der Arbeitsgruppe Nao Team Humboldt. Viele hilfsbereite und interessante Charaktere begleiteten meinen Weg. Claas-Norman Ritter, Thomas Krause, Florian Holzhauer, Paul Schütte, Yuan Xu, Kirill Yasinovskiy und all die anderen, die mich in den Jahren meines Studiums begleitet haben: Dankeschön! Es war eine vielseitige schöne Zeit.

Ganz besonders möchte ich an dieser Stelle Heinrich Mellmann herausheben. Zu den Anfängen meiner Arbeit im Nao Team Humboldt hat er mir weiterführendes Wissen der KI, der Robotik und der Programmierung vermittelt. Später wurde er zu einem wichtigen Diskussionspartner für universitäre, wissenschaftlichen und private Fragen. Vielen Dank für deine Zeit und Geduld.

Danke auch Dir – lieber James – für das Zuhause was du mir mit deiner Tauchbasis geschaffen hast. Der Ausgleich, der mir dadurch geboten wurde war ein Garant für mein Weiterkommen im Studium. Trotz all der spannenden aufregenden Arbeit waren die Ägyptenaufenthalte immer wieder wichtig um die nötige Erholung und Motivation zu tanken. Ich danke dir für deine Unterstützung.

Vor allem aber danke ich meiner Familie. Nur durch Eure Unterstützung konnte diese Arbeit überhaupt erst Wirklichkeit werden. Ich danke Euch.