Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Institut für Automatisierungs- und Systemtechnik
Fachgebiet Systemanalyse
Fachgebiet System- und Steuerungstheorie
Postfach 100565
98684 Ilmenau

Abschlussbericht

BMBF-Verbundvorhaben DeepC

Aktiv
autonomes Unterwasserfahrzeug für große Tauchtiefen

Teilprojekt 6
„Prädiktives Führungssystem“

FKZ
03SX104E

Arbeitsgruppe:
Prof. Dr.-Ing. habil. J. Wernstedt
PD Dr.-Ing. habil. P. Otto
Dipl.-Ing. M. Eichhorn
Dr.-Ing. D. Karimanzira
Dipl.-Ing. T. Liebezeit
Dipl.-Ing. T. Pfützenreuter
Dr.-Ing. V. Zerbe

Oktober 2004

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung und Forschung unter dem Förderkennzeichen 03SX104E gefördert. Die Verantwortung für den Inhalt liegt bei den Autoren.

INHALTSVERZEICHNIS

1 Aufgabenstellung

1.1 Einsatzführungssystem

Das Einsatzführungssystem ist ein zentraler Bestandteil der Software des AUV. Während der Einsatzvorbereitung erhält es den Missionsplan, der vom Teilmodul Einsatzausführung zu realisieren ist. Eine ständige Kontrolle der Ausführung und Ausführbarkeit soll Abweichungen von dem geplanten Einsatz frühzeitig erkennen und eine Umplanung einleiten. Ziel ist eine sichere Erfüllung der Mission oder, wenn die aktuelle Lage keine Alternative zulässt, die Rückkehr zum Rendezvouspunkt für den Notfall.

Folgende Aufgaben sind von der Einsatzführung im einzelnen zu erfüllen:

sequentielle Abarbeitung des aktuellen Missionsplanes (Einsatzausführung)
Kontrolle der Umsetzung der einzelnen Missionsabschnitte (Einsatzkontrolle)
Umplanung der Mission unter den gegenwärtigen Bedingungen (Einsatzumplanung).

Die Umplanung der Mission soll von der Einsatzkontrolle sowie dem Einsatzmanagementsystem veranlasst werden können. Letzteres beinhaltet eine Fahrzeugdiagnose und löst einen Umplanungsvorgang zum Beispiel bei beschränkten Ressourcen (Energie) aus.

Die Umplanung muss gewährleisten, dass das AUV sicher die Mission erfüllen oder im Notfall zu einer Rendezvous–Position gelangen kann.

1.2 Führung des Fahrzeuges in Sondersituationen

Das Fahrzeug arbeitet im Normalfall einen vorgegebenen Missionsplan ab. Dieser Plan enthält Positionswerte und geometrische Daten zur Beschreibung von Basismanövern, welche das Fahrzeug in einer vorgegebenen Reihenfolge abzufahren hat. Tangieren Objekte die Sollbahn des Fahrzeuges, muss in dieser Sondersituation die automatische Führung durch andere Algorithmen abgelöst. In Abhängigkeit der Missionsaufgabe kann die Aufgabe dieser Algorithmen die Führung des Fahrzeuges zum Ausweichen oder zum Identifizieren der Objekte sein.

Das selbständige Vermeiden von Kollisionssituationen ist eine grundlegende Eigenschaft eines AUVs. Im einfachsten Fall bedeutet dies das Umfahren eines ruhenden Objektes. Das AUV muss aber auch in der Lage sein, mehreren, zum Teil bewegten Objekten, sicher auszuweichen. Ein Ausweichen wird notwendig, wenn sich ein Objekt, z. B. ein Wrack oder größere treibende Teile, im Einsatzgebiet befinden. Aufgabe des einzuleitenden Manövers ist es, Kollisionen zu vermeiden und auf dem kürzesten oder sichersten Weg zur Sollbahn zurückzugelangen.

Aufgrund des dynamischen Verhaltens des AUVs (und ggf. der auszuweichenden Objekte), der Umwelteinflüsse (Seeströmung), der begrenzten Reichweiten der Aufklärungs-Sensorik (hier z. B. Forward Looking Sonar) sowie der entfernungsabhängigen Detektionsgüte und Auflösung der Sensoren im Medium Wasser, handelt es sich bei der gestellten Problematik in der Regel um ein sehr komplexes nichtlineares Problem. Die Ausweichstrategien sollten bezüglich dieser o.g. Eigenschaften robust ausgelegt werden.

Als besondere Probleme sind in diesem Zusammenhang die Verallgemeinerung auf (sehr) viele Hindernisse, die zum Teil nicht gleichzeitig im Auffassbereich des Hinderniserfassungssensors liegen, und das Ausweichen in drei Dimensionen zu erwähnen.

Durch die begrenzten Energieressourcen des Fahrzeuges ist eine Auslegung der Ausweichstrategien bezüglich energie- und/oder zeitoptimaler Vorgaben vorzusehen.

Mögliche Sensorausfälle oder defekte bzw. beschädigte Antriebseinheiten sollten in den Ausweichstrategien ebenfalls Berücksichtigung finden.

Eine weitere Sondersituation besteht in der Identifikation von unbekannten Objekten während einer Mission. Eine Identifikation kann eine visuelle oder kartographische Erfassung des Objektes und/oder eine Bestimmung seiner physikalischen/chemischen Eigenschaften beinhalten. Die Steuerung des Fahrzeuges soll hierbei durch Algorithmen zur Positionierung und Führung des Fahrzeuges übernommen werden.

1.3 Maschinelles Lernen

Entwicklung und Test von maschinellen Lernverfahren für AUV´s

Auf dem Gebiet der Künstlichen Intelligenz sind zur Zeit eine Vielzahl von maschinellen Lernverfahren bekannt. Sie unterscheiden sich z. B. durch ihre Zielstellung in induktive, deduktive oder abduktive Lernmethoden. Hinsichtlich ihrer Ergebnisse können Black-Box-Verfahren und Verfahren zum Lernen struktureller Beschreibungen unterschieden werden. Hinsichtlich der Lernstrategie unterscheidet man Lernen durch Beispiele, Lernen durch Analogieschluss, Lernen durch Instruktion und schließlich hinsichtlich der benutzten Methoden statistische, informationstheoretische, genetische, neuronale u. a. Verfahren.

Die Ergebnisse können in Form von Entscheidungsbäumen, Regeln, semantischen Netzen usw. dargestellt werden. Zielsetzung ist es deshalb, zu untersuchen, welche der heute bekannten maschinellen Lernverfahren für das Einsatz- und Manövermanagement am besten geeignet sind und sie den spezifischen Anforderungen von AUV´s anzupassen und zu testen.

Anwendung und Test von maschinellen Lernverfahren für das Manövermanagement

Die autonome Ausführung von Manövern durch ein AUV ist ein sehr komplexer Prozess. Dazu gehören die selbständige Lokalisierung als ein Element der Navigation genauso wie die Fähigkeit, bestimmte Manöver wie Fahren in einer bestimmten Höhe über Grund, Fahren auf einer konstanten Höhe, Entlangfahren an Hindernissen mit einem bestimmten Abstand oder Umfahren eines Hindernisses usw. Das zur autonomen Navigation häufig verwendete „mapping“ der realen Welt mit einer internen Repräsentation soll hier nicht weiter betrachtet werden.

Hier sollen schwerpunktmäßig Methoden des maschinellen Lernens aufbereitet und weiterentwickelt werden, die dem AUV die Fähigkeit verleihen, eine Kollisionssituation erfolgreich zu vermeiden. Im Ergebnis soll das AUV entweder auf den durch die Einsatzplanung vorgegebenen Kurs zurückgeführt werden oder wenn der ursprünglich vorgesehene Einsatzplan nicht mehr realisierbar ist, eine Einsatzumplanung vorgenommen werden.

1.4 MLD – Mission Level Design

Die Sicherstellung des Gesamt-Projektergebnisses durch technisch-/koordinierende Begleitung der Partner unter Systemaspekten ist Kern dieses Arbeitsfeldes.

Eine zielkonforme Durchführung eines komplexen Vorhabens mit vielen Projektpartnern ist eine Aufgabe, die ständige Abstimmung erfordert. Unter Systemgesichtspunkten muss geprüft und bewertet werden, ob jeweils erarbeitete Lösungen zueinander kompatibel sind, ob Optimierungskriterien im Gesamtzusammenhang sinnfällig sind.

Bedingt durch die weitgehend getrennte Entwicklung von Hard- und Software sowie die meist erst zum Integrationstest berücksichtigten Anforderungen seitens der Architektur und der Erfüllung von Echtzeitbedingungen werden Entwurfsfehler erst später erkannt.

2 Voraussetzungen, unter denen das Vorhaben durchgeführt wurde

Das Fachgebiet Systemanalyse [SYS] verfügt seit Jahren über umfangreiche theoretische und praktische Erfahrungen auf den Gebieten:

der Modellierung,
der Simulation,
der Steuerung / Regelung,
des Entwurfs von Entscheidungshilfesystemen für den Menschen,
der Methoden der Computational Intelligence,
des maschinelle Lernens

und deren hard- und softwaretechnischen Umsetzung in komplexen dynamischen Umgebungen in Technik, Medizin und Umwelt. Das theoretische und ingenieurtechnische Wissen wurde insbesondere bis jetzt in Projekten

Während der jahrelangen Arbeit auf dem Gebiet der Fuzzy-Theorie wurden im Fachbereich Softwarewerkzeuge für die Lehre und Forschung entwickelt [KMW93], [DKO97], [EKW97], in denen die aktuellen Projekt- und Forschungsergebnisse einflossen [EICH95]. Durch diese Tools können optimale Fuzzy-Systeme nach nutzerdefinierten Anforderungen erstellt werden. In diesem Zusammenhang entstand auch die Fuzzy Control Design Toolbox für MATLAB® [KMW93].

Zum maschinellen Lernen lagen bereits Erfahrungen vor, die im Rahmen des durch das BMFT geförderten Projektes WISCON (7/91 - 12/94) bei der Wissensakquisition zur Steuerung dynamischer Prozesse (WEDP) gemacht wurden und in deren Ergebnis ein Programmsystem ILMWISSAK entstanden ist. Es ermöglicht eine auf die aktuelle Situation bezogene Prognose zukünftig auszuführender Handlungen, wenn eine bestimmte Zielstellung vorgegeben ist. Die Ermittlung der zukünftigen Handlungsschritte erfolgt dabei durch Anwendung von Inferenzverfahren auf eine mit dem Programmsystem "Fuzzy Opt" [OTT95b], [DKO97] gelernte Regelbasis, wobei durch die Kombination mit Fuzzy-Methoden sehr gute Möglichkeiten der Verarbeitung von unscharfen Informationen gegeben sind.

Verfahren des maschinellen Lernens wurden außerdem bei zahlreichen Praxisprojekten eingesetzt, wie z. B. der Steuerung von Talsperren zur Vermeidung von Hochwassersituationen [KRS94], zur Analyse von Wechselwirkungen bei Biosignalen des Herz- Kreislaufsystems [OM98] und der Beschreibung von Deformationsprozessen bei Talsperren [HEI99].

Künstliche Neuronale Netze stellen seit vielen Jahren einen Lehr- und Forschungsschwerpunkt des Fachgebietes Systemanalyse dar. Hierbei werden vor allem Probleme der Modellbildung, der Steuerung/Regelung und der Klassifikation zur Entscheidungsfindung behandelt [OTT95a],[SEL95].

Das Fachgebiet Systemanalyse arbeitete in einem Industrieprojekt mit dem Projektpartner STN ATLAS Elektronik GmbH bei der Softwareentwicklung zur Simulation und Führung von ROVs zusammen. Diese Arbeit vermittelte ein Verständnis für praktische und theoretische Zusammenhänge in der Unterwassertechnik. So wurden umfangreiche Kenntnisse zum dynamischen Verhalten und zur Führung von Unterwasserfahrzeugen gewonnen.

Die Tätigkeitsfelder des Fachgebietes System- und Steuerungstheorie [THEO] sind

- Mobile Satelliten-Kommunikationssysteme

Seit einigen Jahren arbeitet die Arbeitsgruppe System- und Steuerungstheorie auf dem Gebiet der mobilen Satellitenkommunikation. Hier ist ein Mission Level Design Tool entwickelt worden (Projekt mit Cadence Design Systems), das es zum einen gestattet, die Dynamik, Sichtbarkeit und Abdeckung von Satellitensystemen [SLRG98] zu analysieren und zu animieren, und zum anderen Positionsberechnungen von Konstellationen vorauszuberechnen. In Verbindung mit dem Discrete Event Network Simulator BONeS

Designer sind Trade-off-Untersuchungen bei der Dimensionierung von Satellitensystemen möglich [SLUG98].

- Parallele Systeme

Weiteres "Know-how" war durch die langjährige Forschungsarbeit auf dem Gebiet "Parallele Systeme" und speziell der Parallelisierung komplexer Algorithmen und Implementierung auf verteilten Rechnerarchitekturen vorhanden [HZ94]. Von besonderem Wert ist eine gemeinsame Arbeit mit dem Fachgebiet Neuroinformatik der TU Ilmenau zur Thematik "Merkmalsextraktion aus Bildern und Parallelisierung mit Hilfe Neuronaler Netze" [GH93].

- Mission Level Design (MLD)

Hauptforschungsschwerpunkt der Gruppe ist Mission Level Design. Die Arbeitsgruppe beschäftigte sich mit MLD-Methoden bereits innerhalb des Projektes "Entwicklung von Methoden und Algorithmen zur Echtzeitlagebestimmung und Steuerung mittels GPS" (Thüringen), die es ermöglichen, aus einer Gesamtsystemspezifikation [KRS94] Algorithmen automatisch in Hardware/Software zu übersetzen. Das entwickelte Softwaresystem gestattet die Modellierung von Ressourcen und der Funktion (transformatorisch, reaktiv) für ein Gesamtsystem und die Ausführung der Modelle. In Verbindung mit einer 3D-Visualisierungskomponente sind die Simulationsergebnisse unmittelbar sichtbar. Die weitere Validierung der Algorithmen erfolgte an einem Großflugmodell **[**ZER99].

Die beiden Fachgebiete erstellten im Vorfeld des Projektes DeepC für den Projektpartner STN ATLAS Elektronik GmbH eine Forschungsstudie "Manuelle Fahrzeugführung in virtuellen Welten" [WS99]. Ziel der Studie war es, in einer ersten Stufe auf dem Weg zur Entwicklung eines wissensbasierten Führungssystems für AUVs, eine Softwareumgebung für die Simulation, Darstellung, Analyse und Bewertung einer manuellen Führung von Unterwasserfahrzeugen in virtuellen Welten zu schaffen.

Literatur

[DKO97] DUNG, L.; KOCH, M.; OTTO, P.: *FuzzyOpt – ein Werkzeug zum Entwurf optimaler Fuzzy-Systeme*; at-Automatisierungstechnik 45 (1997) 11, pp 555-556

[EICH95] EICHHORN, M.: *Gütekriterien und Suchverfahren für die Optimierung von Fuzzy Systemen*; Diplomarbeit, TU Ilmenau, 1995.

[EKW97] EICHHORN, M.; KUHN, TH.; WERNSTEDT, J.: *Die Fuzzy Control Design Toolbox für MATLAB*; at-Automatisierungstechnik 45 (1997) 11, pp. 553-554

[FZS98] FREUND, U.; ZERBE, V.; SCHORCHT, G*.; HIFAQ's: a Language for Integrated System Level Simulation*; CESA ’98 IMACS Conference; Nabeul-Tunesien.

[GH93] GRZEMBA, C.; HENKE, K.: *Farming als Methode zur Parallelisierung komplexer Algorithmen der linearen Algebra*. TAT ’93.

[HEI99] HEINE, K.: Beschreibung *von Deformationsprozessen durch Volterra- und Fuzzy-Modelle sowie Neuronale Netze*; Dissertation, Technische Universität Braunschweig, 1999.

[HZ94] HIRSCHFELD. R.; ZERBE, V.: *Untersuchungen und Implementierung von Algorithmen der linearen Algebra auf Systemen mit verteiltem Speicher*. ETF '94, Bielefeld.

[KZTR98] KAYNAK, O.; ZADEH, L. A.; TÜRKSEN, B.; RUDAS, I.J. (eds.): *Computational*

Intelligence: Fuzzy-Neuro Integration with Applications; Springer-Verlag Berlin Heidelberg 1998.

[KRS94] KOCH, M.; RAUSCHENBACH, TH.; SCHEIBEL, T*.: Endbericht zur Entwicklung der Modelle für die Vorhersage des bodennahen Ozons*, Bericht für die Thüringer Landesanstalt für Umwelt Jena, TU Ilmenau 1994.

[KMW93] KUHN, T.; MARQUARDT, R.; WERNSTEDT, J*.; Ilmenauer Fuzzy Tool, Benutzerhandbuch*; TU Ilmenau; SEI GmbH, Ilmenau 1993.

[OTT95a] OTTO, P.: *Identifikation nichtlinearer Systeme mit künstlichen Neuronalen Netzen*; at-Automatisierungstechnik 43 (1995) 2, pp. 62-68.

[OTT95b] OTTO, P*.: Fuzzy Modelling of Nonlinear Dynamic Systems by Inductive Learned Rules*; 3rd European congress on intelligent techniques and soft computing; Aachen, August 1995, EUFIT ’95, Proceedings Volume 2 pp. 858-864.

[OM98] OTTO, P.; MALBERG, H.; *Fuzzy-Modellbildung zur Analyse von Wechselwirkungen bei Biosignalen des Herz- Kreislaufsystems*; 8. Workshop "Fuzzy Control", Dortmund, Berichtsband pp. 82-95, 1998.

[SLUG98]SATLAB Users Guide - Version 4.0. SatLab LLC, Palo Alto, CA, 1998.

[SLRG98]SATLAB Reference Guide - Version 4.0. SatLab LLC, Palo Alto, CA, 1998.

[SEL95] SELLE, T.: *Steuerung eines mechatronischen Systems mit Künstlichen Neuronalen Netzen*. Anwendersymposium MIT Aachen, 13.-15.Dezember 1995 Erfurt, Tagungsband, pp. 116-121.

[SIM99] SIMULINK, Users Manual, The MathWorks, Inc. 1999

[THEO] http://www-sst.theoinf.tu-ilmenau.de/sst/sst_g.html

[SYS] http://www.Systemtechnik.TU-Ilmenau.DE/~fg_sa

[WS99] WERNSTEDT, J.; SALZWEDEL, H.; et al.; *Abschlussbericht zum Projekt "Manuelle Führung von Unterwasserfahrzeugen in virtuellen Welten"*;TU-Ilmenau, 1999

[ZER99] ZERBE, V.; Report: Entwicklung von Methoden und Algorithmen zur Echtzeitlagebestimmung und Steuerung mittels GPS; Januar ’99.

3 Planung und Ablauf des Vorhabens

Die Planung des Verbundprojektes erfolgte in gegenseitiger Abstimmung mit den am Projekt beteiligten Partnern.

Die Aufgabenstellung und der zeitliche Verlauf der einzelnen Bearbeitungsphasen wurden in einer mit allen Partnern abgestimmten Vorhabensbeschreibung zum Verbundprojekt DeepC festgelegt.

Nach Erarbeitung der endgültigen Gesamtkonzeption DeepC hatten sich auch für das Teilprojekt Prädiktives Führungssystem wesentliche Mehraufwendungen ergeben, die durch einen Aufstockungsantrag abgesichert werden mussten. Dadurch ergaben sich auch zeitliche Verschiebungen gegenüber dem ursprünglich geplanten Ablauf des Vorhabens. Der Abschluss des Teilprojektes verschob sich auf den 31.05.2004.

Dieser Termin wurde wie vorgesehen eingehalten.

Inzwischen konnten die ersten Tests der im Teilprojekt Prädiktives Führungssystem erarbeiteten Lösungen erfolgreich bei Flachwasserversuchen in der Ostsee mit einer anderen Testplattform absolviert werden.

Die erreichten Ergebnisse wurden auf den turnusmäßig durchgeführten Gesamtprojektberatungen vorgestellt und auf zahlreichen Tagungen im In- und Ausland sowie in Zeitschriftenbeiträgen veröffentlicht .

Die Teilberichte und die eingehende Darstellung der erzielten Ergebnisse im Abschlussbericht enthalten eine detaillierte Beschreibung der entwickelten Verfahren und deren Einsatz im AUV DeepC.

4 Wissenschaftlicher und technischer Stand, an den angeknüpft wurde

Bei der Bearbeitung der einzelnen Aufgabenstellungen wurde von den Erfahrungen auf dem Gebiet der Modellbildung, der Steuerung und der Entwicklung wissensbasierter Systeme des Fachgebietes Systemanalyse und des Mission Level Design des Fachgebietes System- und Steuerungstheorie sowie dem damaligen Stand der Wissenschaft ausgegangen. Die Grundlage bildeten vor allem umfangreiche Literaturstudien auf den bearbeiteten Gebieten.

Das Einsatz- oder Missionsmanagementsystem stellt die oberste Ebene in der Softwarestruktur des Fahrzeuges dar. Seine Aufgaben bestehen in der Durchführung, der Kontrolle und eventuell der Umplanung einer Mission. Für solch ein komplexes System existieren verschiedenste Ansätze zur Modellierung. Logische Programmiersprachen, Fuzzy Algorithmen, Petrinetze und endliche Automaten stellen neben weiteren derzeitig den aktuellen Stand dar. In die Systeme eingebettet werden auch die Aktionen bei Nichterfüllbarkeit eines Missionsabschnittes, also die Missionskontrolle und -umplanung. Der Vorteil dieser Vorgehensweise liegt in dem einfacheren Aufbau des Einsatzmanagementsystems im Fahrzeug, da die gesamte Intelligenz im Vorfeld geplant wird. Nachteilig wirken sich der höhere Aufwand der Missionsplanung und die Anfälligkeit gegenüber Eingabefehlern aus.

Sehr verbreitet sind Beschreibungen in Form logischer Programmiersprachen (Prolog) und Petrinetze [MHM96], [OPSS96]. In Abbildung 4.1 ist beispielhaft eine verallgemeinerte Mission mit den Missionsabschnitten (Zustände P1 bis P4) sowie den Übergängen zwischen den Zuständen (Transitionen t1 bis t5) als Petrinetz dargestellt. Aus jeder Situation muss mindestens ein Übergang zum Abbruch der aktuellen Mission definiert werden. Dies erfordert sehr viel Überlegung bei der Planung, wenn nicht nur der Abbruch der Mission als einziger Ausweg gesehen werden soll.

Ein ähnliches Verhalten wird unter Verwendung logischer Programmiersprachen oder endlicher Automaten erreicht [GIR98]. Hier werden Übergänge zwischen einzelnen Missionsabschnitten durch Regeln oder Transitionen definiert.

Alle genannten Verfahren beinhalten eine Beschreibung des Missionsplanes und der zugehörigen Alternativen. Ansätze zur eigenständigen Missionskontrolle unter Berücksichtigung der Fahrzeug- und Umgebungsbedingungen sind nicht ersichtlich.

Wegplanungsverfahren für Unterwasserfahrzeuge werden derzeit meist zur Planung kompletter Missionen im Vorfeld des Einsatzes angewandt. Algorithmen des fallbasierten Schließens [VG96] stellen die Trajektorie aus in der Vergangenheit gefahrenen Bahnkurven zusammen. Ein großes Problem ist hierbei, dass über das Operationsgebiet Fahrdaten vorliegen müssen. Fehlen gewisse Abschnitte, so werden diese durch Geradenstücke überbrückt. Genetische Algorithmen zur Bahnplanung werden in [WHI99] und [SS96] für die Off-Line Berechnung eingesetzt; erste Erfahrungen zum On-Line Einsatz sind in [SUG98] zu finden.

Die Hindernisvermeidung ist eine primäre Eigenschaft von autonomen Fahrzeugen. So wird in der Fachliteratur und auf Kongressen eine Vielzahl von Verfahren vorgestellt, die jedoch bislang in keinem Fahrzeug wirklich realisiert sind. Die vorgestellten Verfahren unterscheiden sich nach ihrer Leistungsfähigkeit und ihrem Einsatz. Bei AUVs werden die höchsten Anforderungen an Hindernisvermeidungsalgorithmen gestellt.

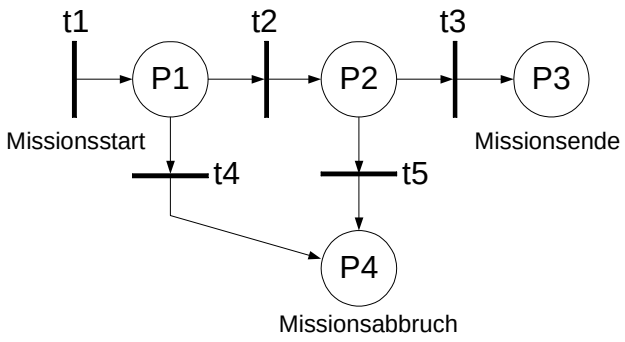


Abbildung 4.1: Petrinetz einer Mission

In FLY95] wird eine hierarchische Struktur für die Hindernisvermeidung verwendet. Die Fuzzy Theorie wird vielfach als Lösungskonzept zur Hindernisvermeidung eingesetzt GVTM99], [BIT], [ZHA94]. In [SE94] wurde eine numerische Lösung zur Ermittlung optimaler Steuertrajektorien beschrieben. Hierbei wurde die Fahrzeugdynamik durch ein mathematisches Modell nachgebildet. Der Einfluss von Umweltbedingungen (Wind, Wellen, Strömung) bei der Generierung von Ausweichmanövern wurde in [XLJY98] berücksichtigt. Ein Ansatz, welcher optimale Manöver auf der Grundlage der Fuzzy-Theorie unter Berücksichtigung der Eigendynamik des AUVs und von Umwelteinflüssen beschreibt, wurde nicht gefunden. Eine Berücksichtigung des möglichen Ausfalls von Sensoren und/oder Aktoren fand bei keinem der bekannten Hindernisvermeidungsalgorithmen für AUVs statt.

Für die reaktive Ebene des Hindernisvermeidungssystems wurde ein neues Verfahren entwickelt, welches auf den Arbeiten von [GUL95], [GUL97] basiert. Das bei diesem Verfahren notwendige Zusammenfassen von Ellipsen zu einer Ellipse mit minimalen Fläche erfolgte durch Algorithmen von [GS98] (siehe Abschnitt). Für die Pfadsuche in der Wegeplanungsebene des Hindernisvermeidungssystems wurde die Boost Graph Library [SLL02], [BOO04] eingesetzt. Sie beinhaltet eine Bibliothek mit fertigen Datenstrukturen für Graphen und Suchalgorithmen und unterstützt eine Vielzahl von C++ Compilern.

Die Bemühungen zum Einsatz von maschinellen Lernverfahren bei der Steuerung von mobilen Robotern und auch AUVs konzentrierten sich in den letzten Jahren schwerpunktmäßig auf den Einsatz Künstlicher Neuronaler Netze [NEH00].

Sie werden sowohl zur automatischen Erkennung der Umgebung (Robot Vision) [KRW] als auch zur autonomen Navigation und Steuerung [TUR95] eingesetzt. Aber auch andere Verfahren des maschinellen Lernens kommen zum Einsatz, wie z.B. das erklärungsbasierte Lernen, induktive (empirische) überwachte, nicht überwachte Lernverfahren [RRS92], hybride Lernverfahren [HSM99] u.a. Diese basieren häufig auch auf der Verwendung von Fuzzy-Systemen und Genetischen Algorithmen (GA) [KZTR98]. Die meisten Arbeiten konzentrieren sich dabei auf landgestützte mobile Roboter.

Veröffentlichungen zum Einsatz von maschinellen Lernverfahren in AUVs, insbesondere zum Einsatz- und Manövermanagement, sind bisher kaum zu finden. Die Möglichkeit der Entwicklung und Erprobung neuer Verfahren auf der Grundlage von Testfahrten in der virtuellen Welt wird bisher offensichtlich kaum genutzt.

Bislang werden komplexe Systeme auf Systemebene entworfen. Dabei werden Hard- und Softwarekomponenten weitgehend getrennt voneinander entwickelt. Verschiedenste Designtools sind für unterschiedlichste Anwendungen verfügbar. Eine ganzheitliche Simulation [BIT] für die Dimensionierung von Systemkomponenten erfordert jedoch den Entwurf auf Missionsebene, auf höherem Entwurflevel. Toolansätze und Ideen für einen auf Missionsebene basierenden Entwurfprozess existieren, Komplettlösungen sind jedoch nicht verfügbar. Hier können Erfahrungen der Gruppe System- und Steuerungstheorie genutzt

werden, die im Bereich Mission Level Design für integrierte Mobilkommunikationssysteme gewonnen wurden [ZFS98], [SLUG98].

Literatur

[BIT] BITTEL O.: Forschungsprojekt Autonomer mobiler Roboter mit neuronaler und unscharfer Steuerung. Fachhochschule Konstanz

[BOO04] http://www.boost.org.

[FLY95] FLYGARE, H.: Collision avoidance for an autonomous underwater vehicle; Master thesis ISRN LUTFD2/TFRT--5545--SE, Department of Automatic Control, Lund Institute of Technology, Lund, Sweden, December 1995.

[GS98] GÄRTNER, B. ; S. SCHÖNHERR S. (1998) *Smallest Enclosing Ellipses An Exact and Generic Implementation in C++* Report B 98-05, Freie Universität Berlin, Institut für Informatik. http://www.inf.fu-berlin.de/inst/ubs/r-b-98-05. abstract.html

[GIR98] GIRARD, A. R*.: A Convenient State Machine Formalism for High-Level Control of AUVs*; Master of Science in Engineering Thesis, Florida Atlantic University, 1998. http://www.oe.fau.edu/~agirard/thesis.html

[GUL95] GULDNER, J*: Intelligentes hierarchisches Regelungskonzept für autonome mobile Robotersysteme*. Düsseldorf, Germany: VDI-Verlag, 1995.

[GUL97] GULDNER, J: *Lokale Kollisionsvermeidung für mobile Roboter mittels künstlicher harmonischer Dipol-Potentiale*. at - Automatisierungstechnik, Volume 45, Issue 01, pp. 24-35 1997.

[GVTM99] GRACANIN, D.; VALAVANIS, K. P.; TSOURVELOUDIS, N. C.; MATIJASEVIC, M.: *Virtual-Environment-Based Navigation and Control of Underwater Vehicles*; IEEE Robotics&Automation Magazine, June 1999

[HSM99] HAMZEI, G.H.; SHAH, D.J.; MULVANEY*: On-line learning of fuzzy decision trees for global path planning*; Engineering Applications of Artificial Intelligence, Vol. 12, Num. 1, Febr. 1999, pp. 93-109, Pergamon.

[KRW] KANADE, T.; REED, M. L.; WEISS, L. E*.: New Technologies and Applications in Robotics*. Communications of the ACM, Band 37, Nr. 3.

[MHM96] MARCO, D.B.; HEALEY, A. J.; MCGHEE, R. B.: *Autonomous Underwater Vehicles: Hybrid Control of Mission and Motion*; Autonomous Robots, vol. 3, Kluwer Academic Publishers, 1996.

[NEH00] NEHMZOV, U.; *Mobile Robots: A Practical Introduction*. Springer-Verlag London Limited 2000.

[OPSS96] OLIVEIRA, P.; PASCOAL, A.; SILVESTRE, C.; SILVA, V*.: Design, Development and Testing of a Mission Control System for the MARIUS AUV*; in Proceedings of the IARP Workshop on Subsea Robotics, 1996.

[RRS92] REISS, K.; REISS, M.; SPANDL, H. (Hrsg.): *Maschinelles Lernen*; Springer-Verlag Berlin Heidelberg, 1992.

[SE94] SPANGELO, I.; EGELAND, O.: Trajectory Planning and Collision Avoidance for Underwater Vehicles Using Optimal Control. IEEE Journal of Ocenic Engineering, Vol. 19, No 4, October 1994

[SLL02] SIEK J.G.; LEE L.-Q. ; A. LUMSDAINE A.: *The Boost Graph Library*: User Guide and Reference Manual. New York, USA: Addison Wesley, 2002.

[SS96] SUGIHARA, K.; SMITH J.: *A Genetic Algorithm for 3-D Path Planning of a Mobile Robot*; Tech. Rep. No. 96-09-01, University of Hawaii, Manoa, 1996

[SUG98] SUGIHARA K.: *GA-based On-line Path Planning for SAUVIM*; In proceegins of the 11th international conference on industrial & engineering applications of artificial

intelligence & expert Systems (IEA-98-AIE); Vol. II; Castellon, Spain, 1998

[TUR95] TURNER, R.M.; *Intelligent Control of Autonomous Underwater Vehicles: The Orca Project*; in proceedings of the 1995 IEEE Conference on Systems, Man and Cybernetics, Vancouver, BC, Canada.

[VG96] VASUDEVAN, C.; GANESAN, K.; *Case-Based Path Planning for Autonomous Underwater Vehicles*; Autonomous Robots, Vol. 3, Kluwer Academic Publishers, 1996.

[WHI99] WATANABE, K.; HASHEM, M. M. A.; IZUMI, K.; *Global Path Planning of Mobile Robots as an Evolutionary Control Problem*; European Control Conference, Karlsruhe, 1999.

[XLJY98] XUEMIN, L.; LIANG, P.; JIAWEI, L.; YURU, X.; *Obstacle Avoidance Using Fuzzy Neural Networks*; UNDERWATER TECHNOLOGY ’98, Tokio

[ZFS98] ZERBE, V.; FREUND, U.; SOLANTI, P.; *Towards animated executable specifications for satellite systems*; ELECTRONICS ’98, 7th international conference Sozopol, Bulgaria

[ZHA94] ZHANG, J; *Ein integriertes Verfahren zur effizienten Planung und Ausführung von Roboterbewegungen in unscharfen Umgebungen*; Dissertation Universität Karlsruhe, 1994.

5 Zusammenarbeit mit anderen Stellen

Eine enge Zusammenarbeit erfolgte vor allem mit den am Vorhaben beteiligten Partnern. Das prädiktive Führungssystem ist einer der Hauptbestandteile der aktiven Autonomie des AUV DeepC, so dass eine besonders intensive Zusammenarbeit mit dem Projektkoordinator ATLAS ELEKTRONIK Bremen notwendig war.

Die von der Universität Karlsruhe entwickelte virtuelle Unterwasserwelt schaffte die Voraussetzungen für die simulative Erprobung der entwickelten Missionsplanungsmethoden und Ausweichstrategien sowie zur Gewinnung von Lerndaten aus Operatorsteuerhandlungen für die reaktive Kollisionsvermeidung mit Hindernissen und der Validierung der daraus maschinell gelernten Regeln.

6 Erzielte Ergebnisse

Formel-Kapitel 6 Abschnitt 1

6.1 Einsatzführungssystem

6.1.1 Aufgaben des Einsatzführungssystems

Das Einsatzführungssystem ist in der obersten Ebene der hierarchisch strukturierten Softwarearchitektur von DeepC angeordnet. Hauptaufgabe der Einsatzführung ist die Überwachung und Umplanung einer Mission, wenn die Rahmenbedingungen dies erfordern.

Das System wurde derartig konzipiert, dass eine Anpassung an andere mobile Systeme möglich ist. Deshalb soll in den nachfolgenden Abschnitten 6.1.2 bis 6.1.5 ein Überblick zu den allgemeingültigen Methoden gegeben werden, die für das System entworfen wurden. Daran schließen sich Aussagen zur konkreten Implementierung auf dem AUV DeepC an.

6.1.2 Konzeption und Aufbau des Einsatzführungssystems

Ausgangspunkt für den Entwurf des Missionsmanagementsystems sind die zu lösenden Aufgaben, die sich aus den bereits im Projektantrag vorgestellten Teilproblemen ergeben:

- Überwachung der Missionsdurchführung und Erkennung von Situationen, die eine Umplanung erfordern,
- Ermittlung der notwendigen Handlungen für die Anpassung des Missionsplanes,
- Durchführung der Modifikation des Missionsplanes.

Zunächst soll eine Analyse der oben genannten Aufgaben durchgeführt werden. Aus der Analyse ergibt sich schließlich die Struktur des Systems entsprechend Abbildung 6.1.

Die Überwachung der Missionsdurchführung und vor allem die Erkennung von Umplanungssituationen erfordern intelligente Verfahren, um mit Hilfe allgemeinen Wissens derartige Situationen zu detektieren oder auch aus der Ähnlichkeit zu bekannten Situationen auf die Notwendigkeit zur Umplanung zu schließen. Auch die Feststellung, welche Veränderungen am Missionsplan durchzuführen sind, ist abhängig von der aktuellen Situation und benötigt entsprechendes Wissen. Deshalb werden diese beiden Aufgaben gemeinsam von einem Modul - der *Missionsüberwachung* - ausgeführt. Sie stellt damit die eigentliche intelligente Komponente des Systems dar. Ergebnis der Überwachung sind ein oder mehrere Umplanungsbefehle. Diese dienen dem Modul *Missionsumplanung* als Eingangsgrößen. Das Modul führt die Befehle nacheinander aus und gibt den modifizierten Plan zurück.

Die verfügbaren Umplanungsbefehle bestimmen in großem Maße den Charakter des Systems. So vereinfacht eine Festlegung nach den konkreten Erfordernissen einer Roboterplattform insbesondere den Aufbau der Missionsumplanung, da die speziellen Bedürfnisse der Plattform direkt berücksichtigt werden können. Nachteilig wirkt sich dabei aus, dass das intelligente Missionsmanagement beim Einsatz auf einem anderen mobilen System möglicherweise neu aufgebaut werden muss. Werden die Befehle hingegen weitestgehend universell definiert, ist unter Umständen ein höherer Aufwand bei der Anpassung des Planes erforderlich. Die Ursache dafür liegt in der Notwendigkeit zur Übersetzung der auszuführenden Planmodifikation in eine Folge allgemeingültiger Befehle. Andererseits ergibt sich durch diese Festlegung als großer Vorteil, dass die Menge an Befehlen für die unterschiedlichen Typen mobiler Systeme und darüber hinaus für auch alle sequentiellen Pläne geeignet ist. Deshalb wurde dieser Ansatz in dem Projekt verwendet (Tabelle 6.1).

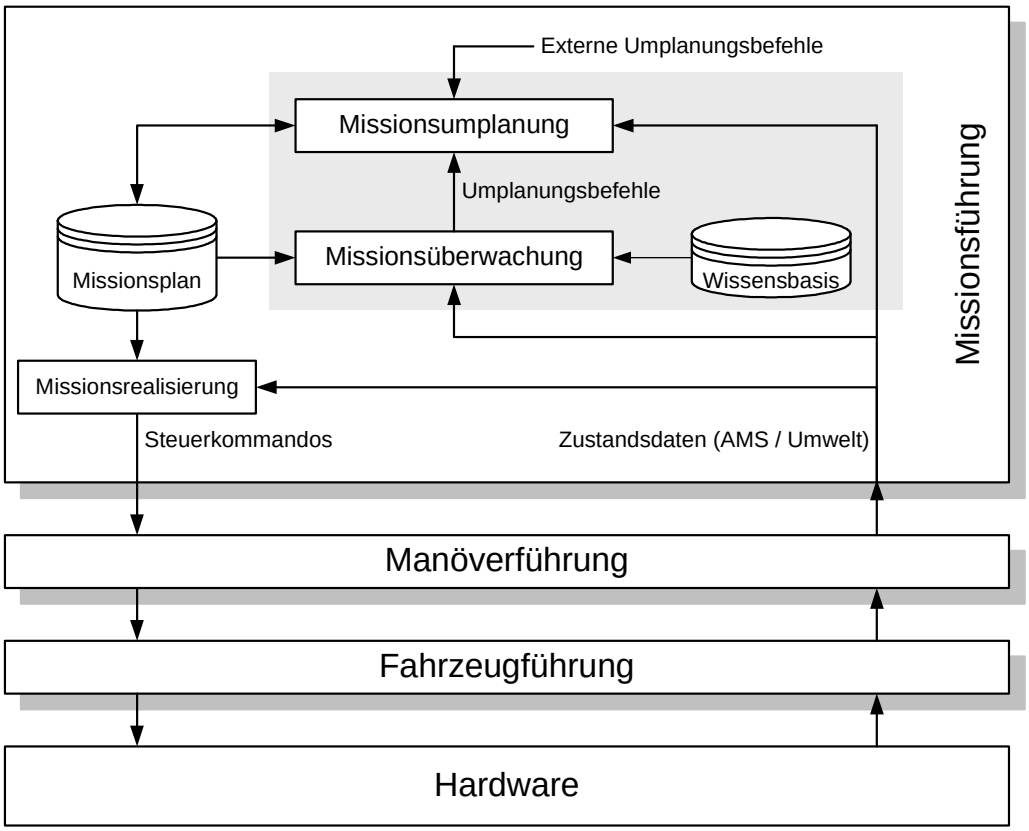


Abbildung 6.1: Aufbau und Integration der Einsatzführung

Tabelle 6.1: Verfügbare allgemeingültige Umplanungsbefehle

Befehl	**Beschreibung**
Atomare Befehle	
Einfügen	Fügt ein Element in den Missionsplan ein. Elemente können sowohl zwischen andere Elemente als auch in ein Element eingefügt werden.
Löschen	Löscht ein Element aus dem Missionsplan.
Modifizieren	Modifiziert Parameter eines Elementes des Missionsplanes.
Komplexe Befehle	
Missionsabbruch	Bricht die Mission ab und fügt die eventuell dafür notwendigen Anweisungen in den Missionsplan ein.
Optimierung	Optimiert einen Missionsplan bezüglich der verfügbaren Ressourcen des autonomen mobilen Systems.

Insbesondere mit den atomaren Befehlen Einfügen, Löschen und Modifizieren sind beliebige Modifikationen an Missionsplänen möglich. Sie stellen die kleinsten, unteilbaren Einheiten für die Veränderung eines Planes dar. Die komplexen Befehle Abbruch und Optimierung setzen sich im Grunde genommen aus den atomaren zusammen. Sie dienen einerseits der Verlagerung umfangreicher Berechnungen in die Missionsumplanung. Dies ist zum Beispiel bei der Optimierung der Fall. Andererseits wird die Übergabe von Umplanungsbefehlen durch andere Komponenten der Systemsoftware vereinfacht. Diese Komponenten können über die gleiche Schnittstelle wie die Missionsüberwachung Befehle erteilen, um eine umgehende Reaktion auf bestimmte Ereignisse auszulösen. Denkbar ist zum Beispiel das sofortige Auslösen eines Missionsabbruchs infolge der Lagebeurteilung durch die Fahrzeugeigendiagnose (Health-Monitoring), ohne extra eine Missionsüberwachung durchzuführen.

Das Missionsmanagement besitzt Zugriff auf alle Zustandsdaten des autonomen mobilen Systems und der Umwelt, die für die Aufgabenerfüllung notwendig sind. Dazu gehören

neben den Navigations- und Sensordaten auch diejenigen abgeleiteten Informationen, die von anderen Systemkomponenten erzeugt werden. Beispielsweise können Diagnosemeldungen wie ein geringer Ladezustand von Batterien zu Veränderungen am Missionsplan, in diesem Fall durch eine Planoptimierung, führen. Mit Hilfe all dieser Daten kann eine Bewertung der aktuellen Situation und gegebenenfalls eine Modifikation des Missionsplanes realisiert werden.

6.1.3 Wissensbasierte Überwachung der Mission

Die wissensbasierte Überwachung ist das zentrale Element des Missionsmanagements, da sie im Wesentlichen die Intelligenz des Systems beinhaltet. Sie wird mit Hilfe eines regelbasierten Systems durchgeführt und ermittelt die in einer bestimmten Situation auszuführenden Veränderungen am Missionsplan.

Im Folgenden sollen die wichtigsten Grundlagen dargelegt werden, die für die wissensbasierte Überwachung der Mission notwendig sind. Nach einem Überblick zu dem allgemeinen Aufbau und der Arbeitsweise regelbasierter Systeme im ersten Teil dieses Kapitels wird das neue Konzept der Missionsüberwachung und seine Integration in das intelligente Missionsmanagement vorgestellt.

6.1.3.1 Aufbau und Arbeitsweise regelbasierter Systeme

Regelbasierte Systeme und besonders regelbasierte Expertensysteme nutzen spezifisches Wissen, um Probleme analog zu einem menschlichen Experten zu lösen. Ein Experte zeichnet sich durch Sachverstand in einem bestimmtes Gebiet aus, der für die Allgemeinheit der Menschen nicht verfügbar ist. Dadurch ist er in der Lage, Probleme einfacher, schneller oder auch genauer zu lösen als jeder andere. Das Wissen eines Experten ist dabei in der Regel auf einen bestimmten Bereich begrenzt (domänenspezifisches Wissen).

Schon frühzeitig wurde erkannt, das dieses Wissen verloren gehen kann. Der „Verlust“ des Experten durch Abwanderung oder Ruhestand ist zum Beispiel bis in die heutige Zeit ein großes Problem für die Industrie. Auch die Nutzung der Erfahrung des Experten an verschiedenen Orten gleichzeitig ist zwar durch moderne Kommunikationsmittel erleichtert worden, ein Ausgleich für die Tätigkeit vor Ort ist dies jedoch nicht. Deshalb besteht seit den 70er Jahren der Wunsch, das Wissen und den Sachverstand von Experten in dauerhafter und universell nutzbarer Form zu hinterlegen. Daraus entwickelten sich die ersten Expertensysteme wie DENDRAL oder MYCIN [GR98]. Heutzutage sind regelbasierte Expertensysteme bedingt durch ihre Vorteile wie Modularität, gute Lesbarkeit sowie einfache Wartbarkeit die weitesten verbreitete Form wissensbasierter Systeme [Moh00].

6.1.3.1.1 Aufbau

Prinzipiell bestehen regelbasierte Expertensysteme aus mehreren Komponenten (Abbildung 6.2).
Die für die eigentliche Aufgabenerfüllung erforderlichen Kernelemente sind [Moh00]:

- *Wissensbasis:* Sie enthält die Regeln und ist im Allgemeinen statisch. Es findet also keine Veränderung der Regeln statt.
- *Arbeitsspeicher:* Er enthält veränderliches Wissen über einen Prozess oder ein System. Häufig wird dieses Wissen fallspezifisch in Form von Fakten hinterlegt.
- *Wissensverarbeitungskomponente:* Sie enthält den Algorithmus zur Ausführung der Regeln. Der dabei ablaufende Prozess wird auch als Inferenz, die Komponente selbst als Inferenzsystem bezeichnet.

Darüber hinaus besitzen Expertensysteme für die Interaktion mit dem Menschen folgende Schnittstellen:

- *Wissenserwerbskomponente:* Sie dient zur Wissensermittlung vom Experten und zur Veränderung des bereits gespeicherten Wissens.
- *Erklärungskomponente:* Sie beschreibt den Weg in Form der aktivierten Regeln, der beim Schließen aufgrund der vorhandenen Fakten beschritten wurde.
- *Dialogkomponente:* Sie dient der Interaktion des Systems mit den verschiedenen Personen (Entwickler, Experte, Nutzer). Darüber wird zum Beispiel der Arbeitsspeicher mit Fakten gefüllt, Regeln eingegeben oder die Steuerung des Systems vorgenommen.

Diese Schnittstellen sind für den Fall, dass ein regelbasiertes System in einem autonomen System eingesetzt werden soll, nur für den Entwurf, die Wissensakquisition sowie zu Test- und Wartungszwecken notwendig. Lediglich Wissensbasis, Arbeitsspeicher und Inferenzsystem werden dann im produktiven Einsatz benötigt. Deshalb sollen die für das Verständnis dieser Komponenten notwendigen Informationen in den nachfolgenden Abschnitten vermittelt werden.

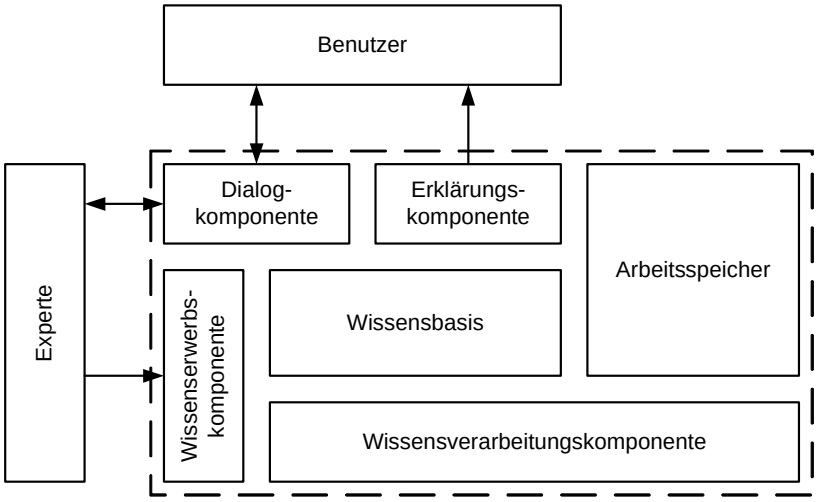


Abbildung 6.2: Aufbau eines regelbasierten Expertensystems

6.1.3.1.2 Wissensrepräsentation

Die Wissensrepräsentation beschäftigt sich mit der Darstellung, Beschreibung und Präsentation von Wissen [Hau00]. Ziel ist es, die Informationen in maschinenverarbeitbarer Form zu speichern. Gleichzeitig soll es möglich sein, bei Bedarf aus der gewählten Beschreibung das Wissen wieder zu reproduzieren. Vorhandenes Wissen kann auf verschiedene Weise in einem wissensbasierten System hinterlegt sein. Prinzipiell werden deklarative und prozedurale Darstellungsarten des Wissens unterschieden.

Die *prozedurale Wissensrepräsentation* beinhaltet Verfahren für die Lösung eines Problems in Form von definierten Abläufen. Das Wissen liegt dabei als Gleichungen, Algorithmen oder Programme vor [BW87].

Die *deklarative Wissensrepräsentation* hingegen stellt die vorhandenen Informationen lediglich als Sachverhalte zur Verfügung, es werden keine Aussagen über die Art der Anwendung des Wissens gemacht. Deklaratives Wissen kann sehr gut in Wissenseinheiten strukturiert werden, die die Basis für einen modularen Aufbau von Expertensystemen bilden. Gebräuchliche Methoden für die Wissensdarstellung sind relationale Datenmodelle und objektorientierte Darstellungen wie semantische Netze, Objekt-Attribut-Wert-Tripel oder Frames [Nik97].

Regelbasierte Systeme und darauf basierende Expertensysteme nutzen Regeln für die Speicherung des deklarativen Wissens. Regeln sind nach [BKI00] formalisierte Konditionalsätze der Form

Wenn *A* Dann *B*,

wobei *A* als Prämisse (oder Antezedenz) und *B* als Konklusion (oder Konsequenz) bezeichnet werden. Da der Konklusionsteil oftmals auszuführende Aktionen beinhaltet, werden derartige Regeln auch als Produktionsregeln und die regelbasierten Systeme als Produktionssysteme bezeichnet. Häufig lassen sich deklaratives und prozedurales Wissen in Produktionssystemen in Kombination anwenden. Dabei können Prozeduren sowohl in der Prämisse als auch in der Konklusion aufgerufen werden. Das erlaubt eine flexible Anpassung an die verschiedenen Formen des vorhandenen Wissens.

6.1.3.1.3 Wissensverarbeitung

Aufgabe der Wissensverarbeitung ist die Anwendung des vorhandenen Wissens, um aus den enthaltenen Informationen Schlussfolgerungen zu ziehen. Dieser Prozess wird auch als Inferenz bezeichnet, die zugehörige Komponente des wissensbasierten Systems als Inferenzsystem. Für das Schließen existieren unterschiedliche Verfahren, um auch mit großen Wissensbasen effektiv arbeiten zu können.
Die Arbeitsweise der verschiedenen Inferenzstrategien lässt sich prinzipiell folgendermaßen darstellen (Abbildung 6.3). Bei Vorliegen neuer Daten muss zunächst diejenige Menge an Regeln (Konfliktmenge) ermittelt werden, die aufgrund dieser Informationen abzuarbeiten ist. Die so genannte Agenda dient als Speicher für diese Menge an aktivierten Regeln. Anschließend erfolgt die sequentielle Ausführung der Regeln der Agenda, die aufgrund ihres Konklusionsteils zu abgeleitetem Wissen und damit zur Aktivierung weiterer Regeln führen kann. Mit ihrer erfolgreichen Realisierung werden aktivierte Regeln aus der Agenda entfernt. Erst wenn die Agenda komplett geleert ist, wird der Inferenzalgorithmus beendet.

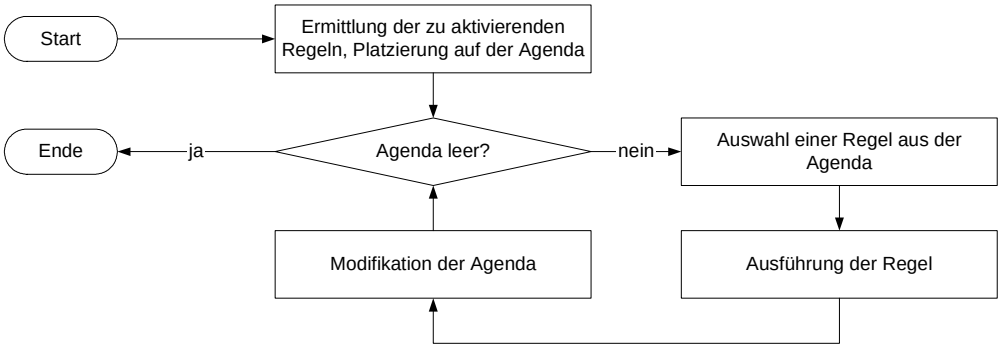


Abbildung 6.3: Ablauf der Inferenz

Mit steigender Regelzahl oder auch wachsendem Faktenwissen über die aktuelle Situation wird die Gesamtlaufzeit der Inferenz im Wesentlichen durch den Prozess der Regelauswahl (Platzierung auf der Agenda) sowie die Modifikation der Agenda beeinflusst. Durch eine Vorverarbeitung des Regelwerkes kann dieser Vorgang jedoch erheblich beschleunigt werden. So generiert der Rete-Algorithmus aus den Prämissen des Regelwerks eine Baumstruktur [For82]. Die einzelnen Bestandteile der Prämissen aller Regeln der Wissensbasis sind darin einmalig hinterlegt und werden nach dem Start des Inferenzverfahrens auf ihre Erfüllung hin untersucht. Das Ergebnis wird in den Knoten des Baumes als Zustände gespeichert. Ändern sich nun Fakten, so werden lediglich diejenigen Knoten neu berechnet, die diese Fakten beinhalten. Bedingt durch die Zustandsspeicherung zwischen den Arbeitszyklen ist der Rete-Algorithmus in Situationen, in denen sich gleichzeitig eine große Zahl von Fakten ändert, nicht effizient. Weiterentwicklungen wie Rete 2 oder Rete/UL bieten sich in derartigen Fällen als praktikable Methoden an [For94], [Doo95].

Vor allem die in den letzten Jahren intensiv untersuchten Echtzeit-Expertensysteme benötigen jedoch eine Inferenzstrategie, die innerhalb einer festgelegten Zeit eine Lösung findet. Die vorgestellten Methoden sind dafür weniger geeignet, da für echtzeitfähige Systeme unumgängliche Eigenschaften wie die Interruptfähigkeit und die Vorhersagbarkeit der Laufzeit fehlen. Deshalb wurden spezielle echtzeitfähige Inferenzverfahren entwickelt. Im Mittelpunkt der Methoden steht wiederum die zeitintensive Ermittlung der Konfliktmenge, die zum Beispiel durch Generierung eines optimalen Entscheidungsbaumes (serielle Faktermittlung) oder durch eine parallele Faktermittlung mit Hilfe von effizienten Regelkodierungen erfolgen kann [Dun95].

6.1.3.1.4 Wissensermittlung

Bevor ein wissensbasiertes System implementiert werden kann, muss zunächst die Erfassung des zu hinterlegenden Wissens erfolgen. Aus dem Blickwinkel des wissensbasierten Systems heißt dieser Prozess *Wissensermittlung* oder *Wissensakquisition*. Wird hingegen primär die Arbeit der wissenserfassenden Person, des Wissensingenieurs, betrachtet, ist der Begriff *Knowledge Engineering* zutreffender. Dieser umfasst neben den reinen Entwicklungsaufgaben auch die insbesondere im produktiven Einsatz wichtigen Wartungstätigkeiten [Hau00].

Schon frühzeitig wurde erkannt, dass die Wissensermittlung den größten Engpass bei der Realisierung eines Expertensystems darstellt [HRWL83]. Neben ihrer hohen Zeitintensität hat sie auch großen Einfluss auf die Qualität des entstehenden Systems. Fehlt zum Beispiel Wissen in der Wissensbasis, weil entweder der Experte oder der Entwicklungsingenieur diese Informationen als allgemein bekannt oder unwichtig ansehen, kann das Expertensystem unter Umständen nicht in der gewünschten Weise arbeiten.

Zur Überwindung dieses fehlerträchtigen Prozesses werden vermehrt Methoden des maschinellen Lernens eingesetzt, die als Ziel einen vollautomatischen Wissenserwerb verfolgen. Sie benötigen jedoch umfangreiches Datenmaterial zur Wissensextraktion und Verifikation des Gelernten, das nicht für alle möglichen Einsatzgebiete von Expertensystemen zur Verfügung steht. Beispielsweise sind maschinelle Lernverfahren zur Erstellung eines Systems für das Katastrophenmanagement denkbar ungeeignet.

Der Prozess der Wissensermittlung wird in unterschiedliche Phasen unterteilt, die die Vorgehensweise des Wissensingenieurs näher beleuchten. *Identifikation* und *Konzeptionalisierung* dienen dem Einstieg in das Problemumfeld. Wesentliche Aufgaben sind hier die Problemdefinition, die Auswahl des einzusetzenden Expertenwissens sowie die Ausarbeitung der grundlegenden Konzepte und Zusammenhänge zur Problembeschreibung. Mit den Phasen *Formalisierung*, *Implementation* und *Test* sind die Bestimmung der Art der Wissensrepräsentation, die Hinterlegung des Wissens in Regeln und die Evaluation des Gesamtsystems verknüpft. Während des Tests ist zu entscheiden, ob die Funktion des Expertensystems der Aufgabenstellung entspricht. Ist dies nicht der Fall, ist über Korrekturen des Regelwerkes (*Verfeinerung*), eine Neustrukturierung des erworbenen Wissens (*Redesign*) oder eine *Neuformulierung* des Gesamtproblems eine Anpassung des Expertensystem vorzunehmen. Diese Tätigkeit hat in enger Zusammenarbeit mit dem Experten zu erfolgen.

6.1.3.2 Konzeption des Missionsüberwachungssystems

Nach dem allgemeinen Überblick zu Aufbau und Entwurf von regelbasierten Expertensystemen soll in diesem Abschnitt das Konzept der wissensbasierten Überwachung der Mission vorgestellt werden.

6.1.3.2.1 Aufgaben der Missionsüberwachung

Grundlegende Aufgabe der wissensbasierten Missionsüberwachung ist die Erfassung von Abweichungen zwischen dem gewünschten und dem realen Zustand des autonomen mobilen Systems. Real bedeutet hier, dass die Zustandsbeschreibung durch einen externen Beobachter (objektive Realität, Wissen) oder mit Hilfe der systemeigenen Sensorik und geeigneten Auswertungsverfahren (subjektive Realität, Wahrnehmung) erfolgt.

Der gewünschte Systemzustand ergibt sich anhand unterschiedlicher, für die erfolgreiche Planerfüllung wesentlicher Kriterien. Diese Merkmale werden aus den einzelnen Manövern des Missionsplanes und den zu erreichenden übergeordneten Vorgaben der Mission abgeleitet.

Die Überwachung beschränkt sich auf die Beobachtung und Kontrolle der durch den Missionsplan vorgegebenen Zielstellungen. Die Diagnose von Defekten des mobilen Systems gehört nicht zu ihrem Aufgabenspektrum, erst die sich daraus ergebenden Auswirkungen auf die Durchführung der Mission werden berücksichtigt. Häufig ist ein speziell auf die verwendete Plattform abgestimmtes Diagnosemodul für die Detektion und Beschreibung von hardwarebedingtem Fehlverhalten verantwortlich. Dessen Ergebnisse fließen dann in die Entscheidungsfindung der Missionsüberwachung mit ein.

Ergebnis der Überwachung soll entsprechend der Gesamtkonzeption neben der Detektion eines Fehlverhaltens des mobilen Systems auch ein Vorschlag zur Überwindung dieser Situation sein. Dieser Vorschlag beinhaltet konkrete Angaben über die notwendigen Anpassungen des Missionsplanes. Die zu lösende Aufgabe der wissensbasierten Überwachung der Mission kann somit zusammenfassend folgendermaßen formuliert werden:

Überwache die Zustandsgrößen des Fahrzeuges sowie die Informationen über die Umgebung und ermittle, ob eine Anpassung des (globalen) Missionsplanes des autonomen mobilen Systems notwendig ist. Bestimme die erforderlichen Korrekturen am Missionsplan.

6.1.3.2.2 Aufbau der Missionsüberwachung

Im Abschnitt 6.1.3.1.1 wurde bereits erwähnt, dass für den Einsatz eines Expertensystems in einem autonomen System lediglich die Kernkomponenten Wissensbasis, Arbeitsspeicher und Inferenzsystem benötigt werden. Dementsprechend ergibt sich der prinzipielle Aufbau der Missionsüberwachung aus Abbildung 6.4. Die verfügbaren Informationen über das mobile System, seine Umwelt sowie die Aufgaben der Mission werden durch die Faktermittlung in den Arbeitsspeicher des Produktionssystems übertragen. Dieser Vorgang muss regelmäßig bei Vorliegen neuer Erkenntnisse wiederholt werden.

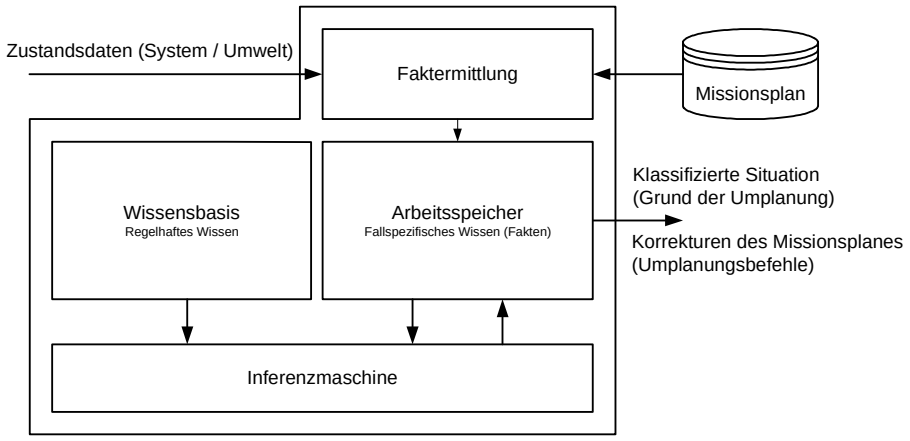


Abbildung 6.4: Aufbau der Missionsüberwachung

Die Arbeit des Inferenzsystems kann zyklisch oder ereignisgesteuert bei Vorliegen von wichtigen Informationen, zum Beispiel dem Ausfall der Sensorik zur Hindernisvermeidung, gestartet werden. Die Produktionsregeln sind derart aufgebaut, dass nach Ausführung des Konklusionsteils Informationen zu den Gründen einer Planmodifikation sowie den durchzuführenden Korrekturen am Plan vorliegen.

6.1.3.2.3 Faktermittlung und Arbeitsspeicher

Die Faktermittlung generiert aus den zur Verfügung stehenden Informationen Fakten, die im Arbeitsspeicher abgelegt werden. Zur Wissensrepräsentation eignen sich besonders objektorientierte Techniken, da sie das Wissen sowie die zur Anwendung notwendigen Verarbeitungsroutinen kapseln.

Die für die Missionsüberwachung wesentlichen Elemente des mobilen Systems wurden zu Objekten zusammengefasst und als Klassen modelliert (Abbildung 6.5). Die Attribute der einzelnen Klassen sind dabei von der jeweiligen Plattform des mobilen Systems sowie der Struktur des Missionsplanes abhängig. Bei der Faktermittlung werden nun Instanzen dieser Klassen erzeugt und mit den verfügbaren Informationen gefüllt. Für einen effektiven Mustervergleich, der die Ermittlung der aktivierten Regeln innerhalb der Inferenzstrategie vornimmt, werden nur diejenigen Informationen in den Arbeitsspeicher eingetragen, die sich gegenüber dem Vorzustand geändert haben. Das reduziert den Rechenzeitaufwand, der entsprechend den Aussagen in Abschnitt 6.1.3.1.3 wesentlich von der Anzahl der geänderten Fakten innerhalb eines Arbeitszyklus des Inferenzsystems abhängig ist.

Die Klasse MONITORING nimmt innerhalb dieser Struktur eine besondere Stellung ein, da sie die Ergebnisse der Missionsüberwachung beinhaltet. Nach Abschluss eines Zyklus der Missionsüberwachung kann über eine Auswertung der Attribute dieser Klasse erkannt werden, ob und welche Modifikationen am Missionsplan durchzuführen sind.

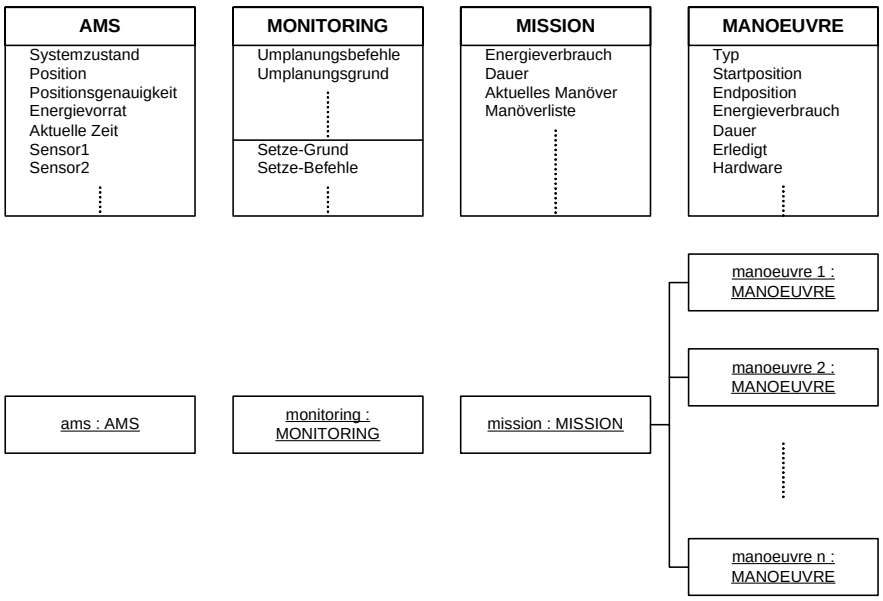


Abbildung 6.5: Klassenstruktur und Objektinstanzen des Arbeitsspeichers

6.1.3.2.4 Wissensbasis

Die Wissensbasis nimmt die während der Wissensermittlung gewonnenen Informationen in Form von Produktionsregeln auf. Sie ist in einzelne Module aufgeteilt. Die Modularität dient der Abgrenzung unterschiedlicher Wissenseinheiten, damit spezielles Wissen für einen bestimmten Typ von mobilen Systemen von dem allgemeingültigen Wissen unterschieden

werden kann. Dadurch ist eine einfache Anpassung der Wissensbasis sowohl an ein anderes mobiles System als auch an weitere Missionsprofile für ein vorhandenes System möglich.

Die Produktionsregeln nutzen die bei der Faktermittlung generierten Objektinstanzen, um die Notwendigkeit zur Umplanung zu bestimmen. In den Prämissen der einzelnen Regeln sind dazu die Bedingungen zu formulieren, die zur Aktivierung der jeweiligen Regeln benötigt werden. Die Tabelle 6.2 zeigt beispielsweise eine Regel zum Entfernen aller Manöver aus dem Missionsplan, die einen bestimmten Sensor (*Sensor 1*) benötigen. Diese Regel verdeutlicht einen wesentlichen Vorteil der regelbasierten Wissensdarstellung gegenüber der prozeduralen Repräsentation. Sind im Missionsplan mehrere Manöver vorhanden, die die Bedingungen der Prämisse erfüllen, wird die Regel für jedes dieser Manöver aktiviert und realisiert. Schleifenläufe wie in prozeduralen Programmiersprachen sind dazu nicht notwendig.

Tabelle 6.2: Regel zum Entfernen von Manövern bei Sensordefekten

```
(defrule HARDWARE:Sensor1-DEFEKT
        ; Priorität der Regel festlegen
        (declare (salience 80))

;=========================================================================
        ; Prämisse

;=========================================================================
        ; Prüfung, ob in der Instanz [ams] Sensor1 als defekt gesetzt ist
        (object (name [ams]) (Sensor1 defekt))
        ; Suchen von Manövern, die die Sensorik Sensor1 verwenden und nocht nicht
erledigt
        ; wurden. In die Variable ?man-name wird der Name des Manövers eingetragen.
        (object (is-a MANOEUVRE) (Erledigt nein) (Hardware $?
"Sensor1" $?)
                (name ?man-name))
=>

;=========================================================================
        ; Konklusion

;=========================================================================
        ; Bildschirmausgabe
        (format t "Lösche Manöver %s wegen Defekt von Sensor1.%n"
?man-name)
        ; Eintragen des Umplanungsgrundes in die Instanz [monitoring] der Klasse
MONITORING
        (send [monitoring] Setze-Grund "Sensor1-Defekt")
        ; Umplanungsbefehl: Löschen des Manövers mit dem Namen ?man-name
        (send [monitoring] Setze-Befehle (str-cat "DEL " ?man-name))
)
```

Die Regeln innerhalb der Wissensbasis können mit Hilfe von Prioritäten anhand ihrer Wertigkeit für die Planerfüllung abgestuft werden. Diese Prioritäten finden innerhalb des Inferenzverfahrens zur Einordnung in die Liste der aktivierten Regeln Verwendung. Damit lässt sich die Reihenfolge der Regelbearbeitung bewusst steuern.

Im Konklusionsteil der Regeln können neue Fakten generiert werden, die der Aktivierung weiterer Regeln dienen. Damit sind auch komplexere Zusammenhänge mit einer größeren Menge einfacher Regeln darstellbar, so dass die Wissensbasis zwar umfangreicher, aber die einzelne Regel verständlicher wird.

Des Weiteren sind derartiger Fakten auch als Status- oder Zustandsinformationen für die Regeln anwendbar. Das ist beispielsweise hilfreich, wenn bestimmte Planmodifikationen nur mit einem gewissen zeitlichen oder räumlichen Abstand zueinander stattfinden sollen (z.B. bei der Stützung der Position des AUV mit einem GPS-Empfänger). Die Auswertung dieser Fakten in der Regelprämisse kann zur Abweisung einer Umplanungsanforderung führen. Die Menge aller Fakten, die als Statusspeicher dienen, repräsentiert dann den internen Zustand des Expertensystems, der während der Missionsdurchführung regelmäßig gesichert werden muss. Damit ist gewährleistet, dass bei einem Neustart des Missionsmanagements infolge eines Rechnerausfalls eine uneingeschränkte Fortsetzung der Arbeit erfolgen kann.

6.1.4 Missionsumplanung

In diesem Kapitel erfolgt die Beschreibung der für die Missionsumplanung benötigten Methoden. Nach einer Einführung in die prinzipielle Arbeitsweise der Umplanung werden die atomaren und komplexen Umplanungsbefehle, ihre Auswirkungen auf den Missionsplan sowie die verwendeten Algorithmen vorgestellt. Die geografischen Verfahren zur Planprüfung und Planmodifikation beschließen die Darlegungen.

Soweit möglich wird bei den Ausführungen von den konkreten Gegebenheiten eines autonomen mobilen Systems abstrahiert. Ein Großteil der Strategien ist nicht auf die Klasse autonomer Systeme beschränkt und besitzen einen breiteres Einsatzgebiet. So wird im Folgenden beispielsweise ein allgemeiner Zustandsvektor für die Beschreibung des Systemzustandes verwendet, der bei einem mobilen System unter anderem aus der Positionsangabe gegenüber einem Referenzsystem besteht.

6.1.4.1 Ablauf der Missionsumplanung

Die Missionsumplanung beginnt mit der Übergabe der zu realisierenden Umplanungsbefehle (Abbildung 6.6). Die Befehle werden zunächst sequentiell ausgeführt. Je nach Art der Planmodifikationen ist es notwendig, im Anschluss den Missionsplan mit Hilfe eines digitalen Geländemodells zu prüfen. So muss zum Beispiel nach dem Einfügen eines neuen Manövers getestet werden, ob das mobile System dieses Planelement befahren kann, ohne die Beschränkungen durch das Gelände zu verletzen.

Die Planprüfung nutzt Routinen zur Schnittberechnung zwischen den einzelnen Planelementen und der geografischen Datenbasis. Dazu wird der Missionsplan in Geradenstücke zerlegt und jede dieser Strecken auf Schnittpunkte mit dem Geländemodell untersucht.

Treten Schnittpunkte auf, soll mit Hilfe der geografischen Planmodifikation für die betroffenen Manöver jeweils eine gültige Route gefunden werden. Dafür ist die geografische Planmodifikation zuständig. Sie nutzt graphenbasierte Suchverfahren, um mögliche fahrbare Strecken zu ermitteln.

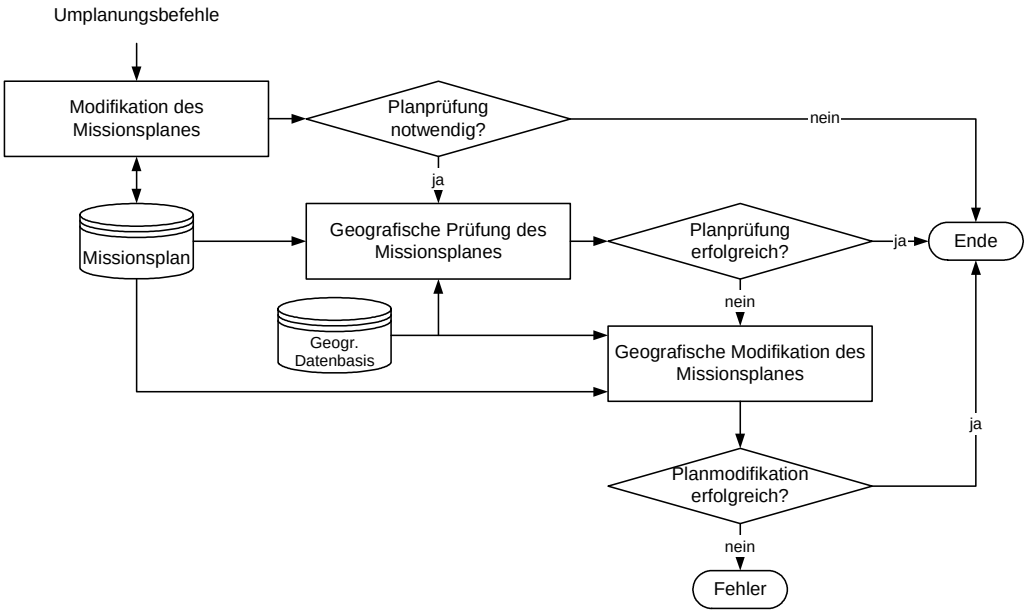


Abbildung 6.6: Grundlegender Ablauf der Missionsumplanung

6.1.4.2 Allgemeine Methoden zur Planmodifikation

6.1.4.2.1 Voraussetzungen

Aufbau des Missionsplanes

Vor der Diskussion der eigentlichen Modifikationsverfahren sollen einige, für das Verständnis wesentliche Grundlagen vermittelt werden. Ausgangspunkt der folgenden Betrachtungen ist die folgende Definition des Begriffes Missionsplan, wie sie für DeepC Anwendung findet:

Definition: Ein *Missionsplan* besteht aus einer Folge von Planelementen. Jedes Planelement stellt dabei ein komplexes Handlungsschema für ein autonomes mobiles System dar und wird auch als Manöver bezeichnet.

Ein Missionsplan ist somit entsprechend Abbildung 6.7 als lineare Sequenz von Planelementen darstellbar. Mit *n* wird dabei die Anzahl der in einem Plan vorhandenen Elemente bezeichnet. Die Festlegung, dass die zu betrachtenden Pläne eine sequentielle Anordnung der Manöver besitzen müssen, schränkt in keiner Weise die Möglichkeiten zur Planmodifikation ein, da mit Hilfe der im Kapitel 6.1.2 definierten Umplanungsbefehle beliebige Veränderungen an einem solchen Plan möglich sind. Ein wesentlicher Vorteil des sequentiellen Planes im Gegensatz zu nebenläufigen oder parallelen Plänen liegt in der Vereinfachung der Planerstellung durch einen menschlichen Operator. Sie weicht zum Beispiel nicht von der Planung des Kapitäns eines Flugzeuges oder Schiffes ab. Die durchzuführenden Aktionen (oder abzufahrenden geografischen Positionen) werden nacheinander geplant und realisiert. Wahrscheinlich gerade deshalb sind sequentielle Missionspläne eine weit verbreitete Planungssprache für autonome Systeme, besonders im Bereich von autonomen Luftfahrzeugen (UAVs, Unmanned Aerial Vehicles) oder autonomen Tauchrobotern (AUVs) [Pri00], [Leo96], [BL99].

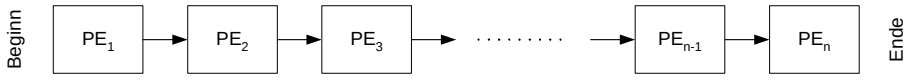


Abbildung 6.7: Aufbau eines sequentiellen Missionsplanes

Eigenschaften von Planelementen

Die einzelnen Planelemente des Missionsplanes werden, wie bereits in der Definition des Begriffs Missionsplan erwähnt, oft auch als Manöver bezeichnet. Deshalb wird dieser Begriff im Folgenden auch als Synonym verwendet. Jedes Planelement besitzt grundlegende Eigenschaften, die in Analogie zu den Operatoren bei den Planerstellungsmethoden der künstlichen Intelligenz benannt werden sollen (Tabelle 6.3) [Her89].

Bevor ein Planelement durch das System aktiviert werden kann, muss der aktuelle Systemzustand dem *Anfangszustand* des Planelementes entsprechen. Je nach Art des autonomen mobilen Systems können unterschiedliche Variablen des Gesamtzustandsraumes für den Zustandsvektor **x** benutzt werden. So wird ein fliegendes oder tauchendes System zumindest die dreidimensionale geografische Position als Beschreibung des Zustandes verwenden. Mit Aktivierung des Planelementes werden dann die damit verbundenen *Aktionen* ausgeführt. Beeinflusst wird deren Arbeitsweise durch den *Parametervektor* **p**. Dieser beinhaltet charakteristische Größen für die unterschiedlichen Manövertypen (zum Beispiel Geschwindigkeiten, Sicherheitsabstände usw.). Eine besondere Rolle innerhalb der Parameter spielen die so genannten Erfüllungsgrade *e*. Davon können bis zu drei implementiert sein:

- Anfangserfüllung e_A - Wert, bei dem die Ausführung des Planelementes beginnt,
- Enderfüllung e_E - Wert, bei dem die Realisierung des Planelementes endet,
- aktuelle Erfüllung e_{akt} - bereits vollendeter Anteil des Manövers, Wertebereich $e_A \leq e_{akt} \leq e_E$.

Die Erfüllungsgrade sind auf die realisierte Zustandsänderung $\Delta\mathbf{x}$ bezogen. Mit Hilfe von Start- und Enderfüllung ist somit eine Korrektur von Anfangs- und Endzustand des Planelementes möglich, die eine große Rolle sowohl bei dem Einfügen-Befehl als auch bei der Optimierung eines Missionsplanes spielt:

$$\Delta\mathbf{x} = \mathbf{x}_E - \mathbf{x}_A$$
$$\mathbf{x}_A^* = \mathbf{x}_A + e_A \cdot \Delta\mathbf{x}$$
$$\mathbf{x}_E^* = \mathbf{x}_A + e_E \cdot \Delta\mathbf{x}.$$

Unter dem Begriff *Ressourcen* werden die bei einem autonomen mobilen System meist begrenzten Vorräte der Manöver, zum Beispiel an Energie oder Zeit zusammengefasst. Der *Wert* eines Planelementes spiegelt den Nutzen wider, den der Systembetreiber bei erfolgreichem Absolvieren des Manövers erhält. Neben den Ressourcen spielt der Wert eine große Rolle bei der Optimierung eines Missionsplanes.

Der *Endzustand* charakterisiert den Zustandvektor bei Beendigung des Planelementes. Er dient als Umschaltkriterium zum nachfolgenden Element im Plan, wobei die Enderfüllung berücksichtigt werden muss. Nicht unerwähnt soll hier bleiben, dass besonderes Augenmerk bei der Realisierung eines solchen Systems auf die ordnungsgemäße Beendigung der gesamten Mission zu legen ist. Durch Einnahme eines sicheren Systemzustandes oder Definition eines Abschlussmanövers kann verhindert werden, dass eine Gefährdung für oder durch das mobile System auftritt. Dies ist sowohl bei der Planung der Mission durch den Operator als auch bei der Umplanung durch das intelligente Missionsmanagement zu berücksichtigen und spielt bei einem etwaigen Missionsabbruch eine große Rolle.

Tabelle 6.3: Eigenschaften von Planelementen

Eigenschaft	Zeichen	Beschreibung
Anfangszustand	$\mathbf{x}_A$	Zustandsvektor bei Aktivierung des Planelements, auch Vorbedingung

Endzustand	$\mathbf{x}_E$	Zustandsvektor bei Beendigung des Planelements, auch Nachbedingung
Aktion(en)	**a**	auszuführende Handlung(en) im Planelement
Parameter	**p**	Parametervektor (z.B. Geschwindigkeiten)
Ressourcen	**c**	benötigte Ressourcen für die Realisierung (z.B. Zeit, Energie), auch Kosten
Wert	w	Erfolgsbewertung des Planelementes, auch Profit

Kontinuitätsbedingung
Mit den genannten Eigenschaften lässt sich ein sequentieller Plan folgendermaßen definieren:

$$\mathbf{x}_A(i+1) = \mathbf{x}_E(i) \quad \forall i = 1,2..,n-1.$$

Dieser auch als Kontinuitätsbedingung bezeichnete Zusammenhang ist die Basis der im Folgenden vorzustellenden Verfahren. Er besagt, dass stets der Endzustand eines Planelementes dem Anfangszustand des nächsten Planelementes entsprechen muss.

Füllplanelemente
Zum besseren Verständnis der nachfolgenden Aussagen soll ein weiterer neuer Begriff, das Füllplanelement, eingeführt werden. Als *Füllplanelemente*} (oder auch Füllmanöver) sollen diejenigen Planelemente bezeichnet werden, die lediglich zum Anpassen von Zustandsunterschieden aufeinander folgender Manöver im Plan vorhanden sind. Sie werden zum Beispiel beim Löschen von Planelementen eingesetzt. Nach Durchführung aller Modifikationen an einem Missionsplan werden aufeinander folgende Füllmanöver in der so genannten Planbereinigung automatisch zusammengefasst. Der Typ und damit die Aktionen eines Füllmanövers hängt von der Wahl des Zustandsvektors für die Beschreibung der Planelemente und damit auch von den konkreten Gegebenheiten der Roboterplattform ab. Für ein mobiles System mit der Fahrzeugsposition als Zustandsvektor besteht zum Beispiel ein Füllplanelement in einem Transit von einer Start- zu einer Endposition, für ein AUV stellen darüber hinaus auch Auf- und Abtauchmanöver Füllplanelemente dar.

6.1.4.2.2 Einfügen von Planelementen

Der atomare Umplanungsbefehl *Einfügen* erlaubt das Hinzufügen von Planelementen zu einem sequentiellen Missionsplan. Er besitzt mehrere zwingend erforderliche Parameter, die den Typ des einzufügenden Planelementes, seine Position im Missionsplan sowie die Art des Einfügens definieren. Dabei wird zwischen dem *Einfügen nach* einem Planelement und dem *Einfügen in* ein Planelement unterschieden.

Das Einfügen nach einem Planelement trennt die Verbindung zwischen den betroffenen Elementen und fügt dazwischen das neue Element ein (Abbildung 6.8). Entsprechend der Kontinuitätsbedingung ergibt sich zunächst, dass Anfangs- und Endzustand des neuen Manövers gleich sind:

vor dem Einfügen: $\mathbf{x}_A(i+1) = \mathbf{x}_E(i)$

nach dem Einfügen:
$$\mathbf{x}_A(E) = \mathbf{x}_E(i)$$
$$\mathbf{x}_E(E) = \mathbf{x}_A(i+1).$$

Je nach Typ des eingefügten Planelementen kann die Gleichheit beider Zustände zulässig sein, ansonsten muss mit Hilfe von Füllmanövern eine Anpassung vorgenommen werden. Dabei kann sowohl ein Füllmanöver vor dem einzufügenden Planelement als auch danach notwendig sein. Das hängt jedoch von der konkreten Implementierung auf dem jeweiligen autonomen mobilen System ab.

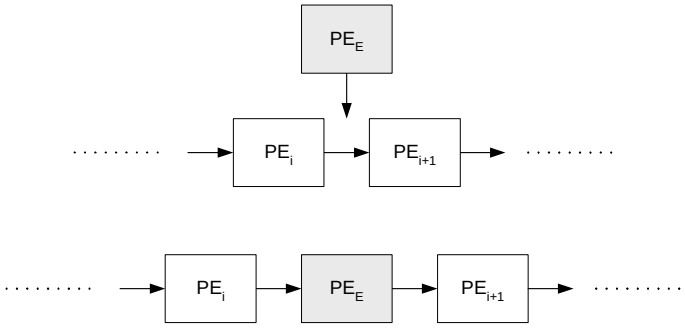


Abbildung 6.8: Einfügen des Planelementes *E* nach dem Element *i*

Beim Einfügen in ein Element wird die Arbeit an einem Manöver unterbrochen, das eingefügte Element realisiert und anschließend das unterbrochene an der gleichen Stelle fortgesetzt, an der es beendet wurde. Um dies in einem sequentiellen Plan darstellen zu können, muss eine zweite Instanz des Planelementes *i* nach dem Element *E* eingefügt werden (Abbildung 6.9). Der im Moment des Abbruchs vorhandene aktuelle Erfüllungsgrad e_{akt} von PE_i^1 wird als Starterfüllung e_A für die Kopie PE_i^2 eingesetzt, so dass dessen Startposition mit der Unterbrechungsposition der ersten Instanz des Manövers identisch ist. Auch hier kann es notwendig sein, dass Füllmanöver vor und nach dem eingefügten Planelement in den Missionsplan aufgenommen werden müssen.

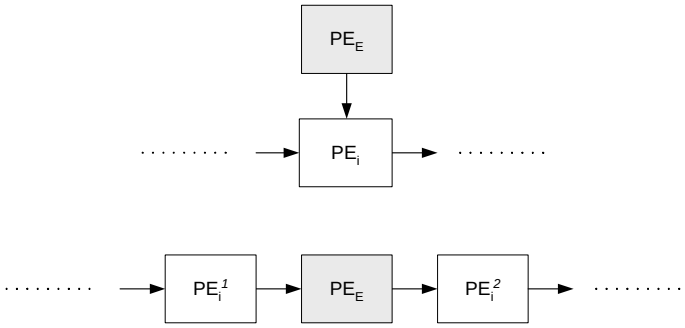


Abbildung 6.9: Einfügen des Planelementes *E* in das Element *i*

6.1.4.2.3 Löschen von Planelementen

Beim Löschen wird das betroffene Manöver aus dem Missionsplan entfernt. Häufig ist daraufhin die Kontinuitätsbedingung verletzt, da für die meisten Planelemente $\mathbf{x}_A(i+2) \neq \mathbf{x}_E(i+2)$ gilt. Um die Kontinuität wiederherzustellen, wird an die Position des gelöschten Manövers ein Füllplanelement eingefügt (Abbildung 6.10). Bei einem mobilen System muss beispielsweise nach dem Löschen eines Planelementes ein Transit von dessen Anfangs- zur Endposition eingefügt werden. Über den Typ des Transitmanövers entscheidet die zu überbrückende Differenz zwischen den beiden Positionen.

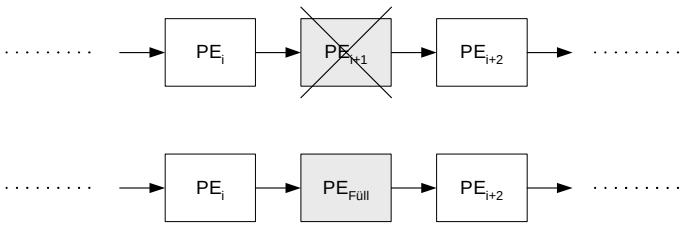


Abbildung 6.10: Löschen des Planelementes *i*

6.1.4.2.4 Modifizieren von Planelementen

Zahlreiche Situationen erfordern eine Anpassung von Parametern eines Planelementes. Dies geschieht mit dem atomaren Befehl *Modifizieren* (Abbildung 6.11). Dabei können die Einflussmöglichkeiten auf den Parametervektor je nach Implementation unterschiedlich sein. So ist es zum Beispiel häufig sinnvoll, die Fahrgeschwindigkeit einzelner Planelemente auf die vorhandenen Ressourcen abzustimmen, während Anpassungen von einem eventuell in den Manöverparametern vorhandenen Prioritätswert meistens nicht notwendig sind.

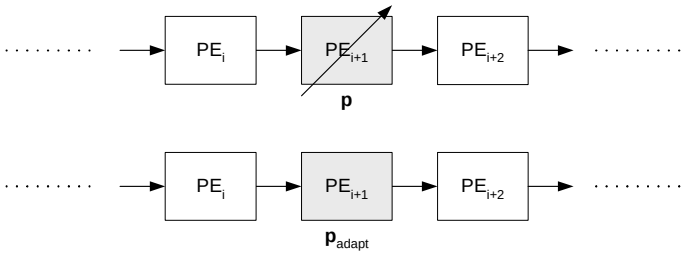


Abbildung 6.11: Modifizieren des Planelementes *i+1*

6.1.4.2.5 Planbereinigung

Ziel der Planbereinigung ist es, aufeinander folgende Füllplanelemente aus dem Plan zu entfernen. Das ist möglich, da diese jeweils nur Zustandsübergänge realisieren, die auch durch ein kombiniertes Manöver der gleichen Art erreichbar sind. Abschnitt \ref{DeepCReplanning} geht auf Besonderheiten der Planbereinigung ein, die sich darüber hinaus für ein im dreidimensionalen Raum bewegendes mobiles System ergeben. Zur Bereinigung des Planes sind mehrere Arbeitsschritte notwendig:

- Bestimmung der Positionen aller Füllmanöver im Missionsplan,
- Ermittlung aufeinander folgender Füllplanelemente (Startindex und Anzahl der Elemente jeder Folge),
- Löschen der auf den jeweiligen Startindex folgenden Füllplanelemente.

Als Ergebnis entsteht ein Missionsplan, der keine aufeinander folgenden Füllmanöver mehr enthält (Abbildung 6.12). Die Planbereinigung wird stets nach dem Löschen oder Einfügen eines Planelementes und während der Optimierung angewendet, da während dieser Umplanungsmaßnahmen Folgen von Füllplanelementen auftreten können.

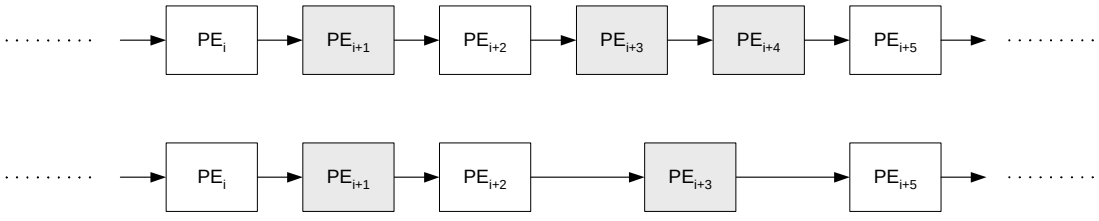


Abbildung 6.12: Bereinigung eines Missionsplanes, Füllplanelemente sind grau hinterlegt

Am konkreten Beispiel eines mobilen Systems wird der Nutzen dieses Verfahrens am einfachsten deutlich. Die in der Abbildung 6.12 grau hinterlegten Füllplanelemente stellen auf einer derartigen Plattform jeweils ein Transitmanöver dar, das Anfangs- und Endzustand miteinander verbindet. Zwei aufeinander folgende Elemente wie PE_{i+3} und PE_{i+4} stellen dabei nicht zwangsläufig den kürzesten Weg dar, um zum Startzustand von PE_{i+5} zu gelangen. Durch die Zusammenfassung beider Füllplanelemente wird eine Wegverkürzung erreicht, die besonders bei der Optimierung eines Missionsplanes einen wesentlichen Beitrag zur Verringerung der benötigten Ressourcen des Planes leistet.

6.1.4.3 Spezielle Verfahren zur Planmodifikation

Die in diesem Abschnitt vorzustellenden Methoden nutzen die allgemeingültigen Verfahren aus Abschnitt 6.1.4.2. Sowohl die dort aufgeführten Aussagen als auch die getroffenen Einschränkungen gelten damit ebenfalls für den Missionsabbruch und die Optimierung von Missionsplänen.

6.1.4.3.1 Missionsabbruch

Der Missionsabbruch stellt einen wichtigen, wenn nicht gar den wichtigsten Umplanungsbefehl dar. Damit soll ein sicherer Abschluss der Mission gewährleistet werden, wenn eine Fortsetzung nicht mehr möglich ist.

Die Ursachen für einen Missionsabbruch können vielfältig sein. Zu nennen sind unter anderem der Ausfall von Sensorik zur Hinderniserkennung, der Ausfall von Aktorik sowie ein extrem schneller Energieabfall. In solchen Situationen ist es erforderlich, die reguläre Planrealisierung zu beenden und stattdessen eine vordefinierte Folge von Planelementen abzuarbeiten, um die Sicherheit des Systems zu gewährleisten.

Voraussetzung für die erfolgreiche Durchführung des Abbruchs einer Mission ist die Verfügbarkeit der dafür notwendigen Hard- und Software des autonomen mobilen Systems. Sind die notwendigen Module nicht funktionsfähig, sollte auf eine, unabhängig von der Steuerung des Systems funktionsfähige Sicherheitseinrichtung zurückgegriffen werden können. Ein AUV benötigt dafür ein Notauftauchsystem, um genügend Auftrieb für das Erreichen der Wasseroberfläche zu erzeugen. Üblich ist die Mitnahme von Ballast, der in Notsituationen abgeworfen werden kann.

Der geplante Missionsabbruch führt demgegenüber bei Funktionsfähigkeit der benötigten Module folgende Schritte aus, die einen sicheren Systemzustand garantieren (Abbildung 6.13):

- Entfernen aller auf das aktuell ausgeführte Planelement folgenden Elemente,
- Einfügen der für einen Missionsabbruch definierten Manöver PE_{Ai}, eventuell Parametrieren dieser Manöver.

Je nach Startzustand des ersten Planelementes eines Missionsabbruchs PE_{A1} kann es erforderlich sein, ein Füllplanelement davor einzufügen, um den Systemzustand zum Abbruchzeitpunkt daran anzupassen. Zum Abschluss dieser Umplanung wird das erste neu eingefügte Manöver aktiviert und vom Führungssystem des autonomen mobilen Systems anschließend realisiert.

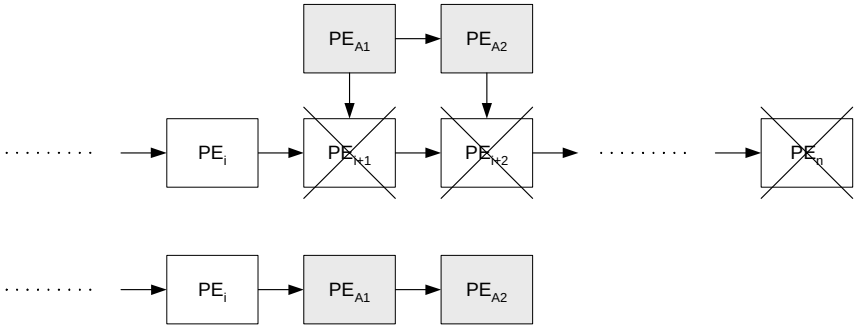


Abbildung 6.13: Abbruch einer Mission bei der Ausführung des Planelementes *i*

6.1.4.3.2 Optimierung eines Planes

Eine Optimierung des Missionsplanes wird durch die Missionsüberwachung angefordert, wenn die vorhandenen Ressourcen des mobilen Systems nicht für die Erfüllung der verbleibenden Planelemente ausreichen. Das Ziel der Optimierung ist, den Plan derart zu modifizieren, dass er sicher verwirklicht werden kann und dabei den größten realisierbaren Nutzen (oder Wert) *W* der Mission repräsentiert. Diese Art der Optimierung versucht demnach, einen bestehenden Plan an die neue Situation anzupassen. Damit soll sichergestellt werden, dass die im Vorfeld bei der Planung durch den Menschen vorgegebene Zielstellung der Mission weitestgehend erhalten bleibt.

Optimierungskriterien und -variablen
Ausgangspunkt für die Lösung einer Optimierungsaufgabe ist stets die Definition des Problems. Wie bereits erwähnt, ist das Ziel dieser Plananpassung, den größtmöglichen Nutzen aus der Mission zu ziehen. Der Wert *W* kann mit Hilfe der einzelnen Planelemente folgendermaßen dargestellt werden:

$$f(\mathbf{x}) = W(\mathbf{x}) = \sum_{k=i}^{n(\mathbf{x})} w_k \cdot x_k \ ,$$

wobei **x** der Vektor der Optimierungsvariablen und *i* der Index des aktuellen Manövers sind. Es ist zu beachten, dass die Anzahl der Planelemente eines Missionsplanes von den Optimierungsvariablen abhängen kann, was durch *n*(**x**) repräsentiert wird. Das zunächst unbeschränkte Optimierungsproblem ergibt sich damit zu:

$$\max\left\{ f(\mathbf{x}) \right\} .$$

Als Beschränkungen für diese Aufgabe treten die verfügbaren Ressourcen des mobilen Systems
$\mathbf{c}_{max}$ auf. Die benötigten Vorräte an Betriebsmitteln sind für jedes Planelement im Vektor **c** hinterlegt, für den Gesamtaufwand **C**(**x**) der verbleibenden Planelemente der Mission gilt dann:

$$\mathbf{C}(\mathbf{x}) = \sum_{k=i}^{n(\mathbf{x})} \mathbf{c}_k \left(x_k \right) .$$

Das Funktional der Beschränkungen kann nun folgendermaßen dargestellt werden:

$$\mathbf{g}(\mathbf{x}) = \mathbf{C}(\mathbf{x}) - \mathbf{c}_{max} = \sum_{k=i}^{n(\mathbf{x})} \mathbf{c}_k \left(x_k \right) - \mathbf{c}_{max} \ .$$

Aus dieser Beziehung wird ersichtlich, dass zum Erfüllen der Beschränkungen $\mathbf{g}(\mathbf{x}) \leq 0$ gelten muss. Das vollständige, beschränkte Optimierungsproblem ergibt sich damit zu:

$$\max\left\{ f(\mathbf{x}) : \mathbf{g}(\mathbf{x}) \leq 0 \right\} .$$

Die Wahl der Optimierungsvariablen beeinflusst wesentlich die Art und den Ablauf der Optimierung. Deshalb müssen vor der Auswahl des zu nutzenden Verfahrens diejenigen Größen festgelegt werden, über die ein Missionsplan an die konkrete Situation angepasst werden kann. Dafür bieten sich verschiedene Parameter an, die direkten Einfluss auf die benötigten Ressourcen des Missonsplanes eines autonomen mobilen Systems haben:

- Über eine *Auswahl der zu realisierenden Planelemente* kann der Betriebsmittelverbrauch mit den verfügbaren Vorräten abgeglichen werden. Auswahl, oder besser gesagt *Aktivierung*, bedeutet hier, dass für jedes verbleibende Planelement entschieden wird, ob es verwirklicht wird. Diese Entscheidung wird mit Hilfe einer booleschen Variable mit dem Wertebereich {0,1} repräsentiert.
- Über die *Parameter der einzelnen Planelemente* kann unter Umständen ebenfalls der Ressourcenbedarf von Manövern reduziert werden. Für ein mobiles System würde sich dafür beispielsweise die Fahrgeschwindigkeit anbieten. Sie beeinflusst über die Leistungs-Geschwindigkeits-Kennlinie den Energieverbrauch sowie über die direkte Verknüpfung mit der Fahrzeit den Zeitaufwand der Gesamtmission. Parameter dieser Art besitzen meist einen kontinuierlichen Wertebereich $p_{min} \leq p \leq p_{max}$.

Bei alleiniger Betrachtung der *Aktivierung der Planelemente* ergibt sich ein diskretes Optimierungsproblem. Diese sind dadurch gekennzeichnet, dass der zulässige Bereich aus einer endlichen oder abzählbar unendlichen Menge *M* besteht [BG93]. Für den hier dargestellten Fall ist *M* als Menge von binären Vektoren definiert: $M \subseteq \{0,1\}^n$. Da sich die Anzahl der Elemente von *M* durch Kombinatorik ermitteln lässt, wird häufig auch von kombinatorischer Optimierung gesprochen. Für einen Satz von *n* Aktivierungswerten ergibt sich 2^n als Mächtigkeit von *M*.

Werden die *Parameter der einzelnen Planelemente* als Variable für die Optimierung näher untersucht, so treten verschiedene Probleme auf. Zunächst ist die Möglichkeit der Einflussnahme auf den Missionsplan durch Kenngrößen einzelner Manöver abhängig von dem konkreten Aufbau und den Aufgaben des mobilen Systems. Es kann also nicht generell davon ausgegangen werden, dass solche Parameter existieren. Andererseits erhöht der meist kontinuierliche Wertebereich solcher Kenngrößen den Aufwand zur Lösung des Problems, so dass durch Nutzung von Manöverparametern als Optimierungsvariablen zusätzlich zur Aktivierung der Planelemente aus einem diskreten ein gemischt ganzzahliges Optimierungsproblem wird. Zugunsten der Allgemeingültigkeit des zu entwerfenden Verfahrens soll aus den genannten Gründen nur die Aktivierung der einzelnen Planelemente als Variable der Optimierung herangezogen werden.

Analogiebetrachtungen und Lösungsansätze

Probleme der im letzten Abschnitt geschilderten Art treten in der Realität häufig auf und stellen deshalb einen wesentlichen Forschungsschwerpunkt der Mathematik und Theoretischen Informatik dar. Ein Standardproblem, an dem oft unterschiedliche Algorithmen der diskreten Optimierung getestet werden, ist das so genannte Rucksack-Problem. Es weist eine große Ähnlichkeit mit der zu lösenden Optimierungsaufgabe auf und soll deshalb im Folgenden näher betrachtet werden.

Speziell das eindimensionale lineare Rucksack-Problem ist zum Vergleich sehr gut geeignet und geht von folgender Situation aus. Es sind *n* verschiedene Gegenstände verfügbar, wobei jeder einen Wert w_k und ein Gewicht (gleich Kosten) c_k besitzt. Ziel ist, diejenigen Artikel auszuwählen, die in einen Rucksack mit dem zulässigen Gesamtgewicht c_{max} passen und dabei den maximalen Gesamtwert W_{max} erzielen. Steht nun jeder Gegenstand nur ein Mal zur Verfügung, so ergibt sich das so genannte 0-1 Rucksack-Problem in der folgenden Weise:

$$\text{Maximiere} \qquad W(\mathbf{x}) = \sum_{k=1}^{n} w_k \cdot x_k$$

$$\text{unter den Nebenbedingungen:} \qquad \sum_{k=1}^{n} c_k \cdot x_k \le c_{max}$$

$$x_k \in \{0,1\} \qquad k = 1,\ldots,n \ .$$

In der Praxis treten derartige Aufgaben häufig auf, so zum Beispiel bei der Finanzplanung[1] oder Projektauswahl[2] [Dom95]. Obwohl dies das einfachste (beschränkte) diskrete Optimierungsproblem darstellt, gehört es zur Komplexitätsklasse NP [GJ79]. Im Gegensatz zur Klasse P der effizient lösbaren Probleme, für die Algorithmen in einer durch ein beliebiges Polynom begrenzten Rechenzeit ein Ergebnis liefern, sind NP-Probleme lediglich effizient überprüfbar. Das Rucksackproblem ist darüber hinaus ein Vertreter der NP-vollständigen Probleme [Lag96]. Derartige Aufgaben gehören zu den schwierigsten der Problemklasse NP und benötigen exponentielle Zeit zur Lösungsfindung.
Durch Anwendung von Heuristiken und Approximationsverfahren mit annähernd optimalen Ergebnissen kann jedoch die Suche nach eine Lösung für viele dieser Probleme stark vereinfacht werden [ACG+99].

Gerade für das Rucksack-Problem existieren effiziente Methoden, die eine Lösung für eine überschaubare Menge von Variablen (oder Gegenständen) mit vertretbarem Zeitaufwand liefern [MT90], [Mur95]. Dazu gehören unter anderem Verfahren der Dynamischen Programmierung, Branch-and-Bound, Tabu Search oder auch Suchverfahren wie Genetische Algorithmen.

Das Optimierungsproblem für die zu lösende Aufgabe besitzt das gleiche Gütefunktional wie das Rucksack-Problem. Die Nebenbedingungen unterscheiden sich jedoch grundsätzlich. Während das Rucksack-Problem lineare eindimensionale Bedingungen besitzt, stellt die in dieser Arbeit zu lösende Aufgabe ein nichtlineares mehrdimensionales Optimierungsproblem dar. Das ergibt sich aus der Ermittlung des Kostenvektors eines Planelementes, da die Nichtrealisierung (x_k=0) in Analogie zum Löschen eines Planelementes meist ein Füllplanelement erfordert:

$$\mathbf{c}_k(x_k) = \begin{cases} \mathbf{c}_k & (x_k = 1) \quad \text{Kosten des Planelementes} \\ \mathbf{c}_{k\,\mathrm{FPE}} & (x_k = 0) \quad \text{Kosten des Füllplanelementes} \ . \end{cases}$$

Wird weiterhin berücksichtigt, dass eine Planbereinigung entsprechend Abschnitt 6.1.4.2.5 im Anschluss an die Zusammenstellung des sich durch den Optimierungsvektor **x** ergebenden Missionsplanes durchgeführt wird, so können die Kosten für die Nichtrealisierung eines Manövers ohne Beachtung der vorhergehenden und nachfolgenden Planelemente nicht bestimmt werden. Das liegt daran, dass durch ein mögliches Zusammenfassen mehrerer Füllplanelemente zu einem kombinierten Element die Kosten nicht mehr nur vom betrachteten Manöver abhängen.

Aus diesem Grund sind die für eine Lösung des linearen Rucksack-Problems verfügbaren Verfahren nicht auf diese Problemklasse übertragbar. Es existieren zwar auch Ansätze für nichtlineare Rucksack-Probleme wie zum Beispiel in [HJM93], doch sind diese wiederum auf spezielle Arten der Nichtlinearität zugeschnitten. Alle anderen Fälle werden häufig mit Hilfe von Suchverfahren oder angepassten Heuristiken gelöst.

Lösung des Optimierungsproblems

[1] Aufteilen einer Geldmenge auf unterschiedliche Anlagen
[2] Entscheidung für die Realisierung bestimmter Projekte aus einer verfügbaren Menge

Ausgangspunkt der folgenden Betrachtungen sind die Einsatzbedingungen des zu implementierenden Verfahrens. Es soll auf den Rechnern des AUV *DeepC* eingesetzt werden und dabei in akzeptabler Zeit eine Lösung präsentieren. Die Anzahl der zu berücksichtigenden Planelemente an sich ist aufgrund unterschiedlicher Überlegungen bereits qualitativ beschränkt:

- Die Anzahl der Planelemente ist durch die Vorgehensweise bei der Missionsplanung begrenzt. Dies liegt an der manuellen Definition der Mission durch den Menschen, der bestrebt ist, die Menge der sequentiell zu planenden Handlungen überschaubar zu halten.
- Eine Optimierung des Missionsplanes wird normalerweise nicht gleich zu Beginn der Mission durchgeführt. Erst nach einiger Zeit mit einem gegenüber der Planung ungleich höheren Ressourcenverbrauch wird eine Anpassung des Planes erforderlich. Zusätzlich werden gewöhnlich bereits bei der Missionsplanung gewisse Reserven an Betriebsmitteln vorgesehen, so dass die Anzahl der Manöver bei Start einer Optimierung geringer als die ursprüngliche Gesamtzahl im Missionsplan sein wird.
- Planelemente, die nur eine Zustandsänderung $\mathbf{x}_A \rightarrow \mathbf{x}_E$ bewirken, stellen laut Definition Füllplanelemente dar. Es kann davon ausgegangen werden, dass derartige Manöver in jedem Missionsplan aufgefunden werden. Diese Füllmanöver können bei der Optimierung unberücksichtigt bleiben, da ihre Nichtrealisierung zum Ersetzen durch ein (gleichartiges) Füllplanelement führt. Auf das Ergebnis der Optimierung hat die Beachtung dieser Manöver keinen Einfluss, sie erhöht lediglich die Komplexität des Gesamtproblems.

Daraus lässt sich jedoch keine quantitative Obergrenze für die Größe des Optimierungsvektors ableiten. Aus diesem Grund wurden zwei Strategien zur Problemlösung entworfen. Während die erste Verfahrensweise bei einer geringen Anzahl von Planelementen zur Optimallösung führt, gewährleistet die für höhere Manöverzahlen entworfene einfache Heuristik eine schnelle Ermittlung eines Ergebnisses. Sie garantiert zwar nicht das mögliche Optimum, liefert aber unter den gegebenen Bedingungen akzeptable Resultate. Ein Vergleich beider Verfahren wird für das AUV DeepC im Abschnitt 6.1.6.4 durchgeführt.

Auffinden der Optimallösung

Zunächst muss der Vektor der Optimierungsvariablen **x** ermittelt werden. Er setzt sich aus der Aktivierung akt_k derjenigen Planelemente zusammen, die keine Füllmanöver darstellen:

$$\mathbf{x} = \mathbf{akt} = \left[akt_i, akt_{i+1}, \ldots, akt_m \right]$$
$$\text{mit} \qquad akt_k \in \{0,1\} \qquad k = [i, \ldots, m]$$

wobei *i* das aktuelle Planelement und *m* der Index des letzten zu berücksichtigenden Manövers ist.

Zur Optimallösung führt ein Enumerationsverfahren, das eine vollständige Aufzählung aller möglichen Kombinationen der Einzelwerte von **akt** generiert. Jeder Aktivierungsvektor repräsentiert dabei eine mögliche Manöverkonfiguration, die auf ihren Wert und ihre Kosten untersucht werden muss. Für eine effiziente Arbeitsweise erfolgt dies in einem zweistufigen Prozess. Während die Ermittlung des Wertes einer Konfiguration einfach durch Bildung des Skalarproduktes $W = \mathbf{w} \cdot \mathbf{akt}$ geschieht, muss für die Kostenberechnung die Manöverkonfiguration durch Löschen der nicht zu realisierenden Planelemente und anschließende Planbereinigung erzeugt werden (Abbildung 6.14).

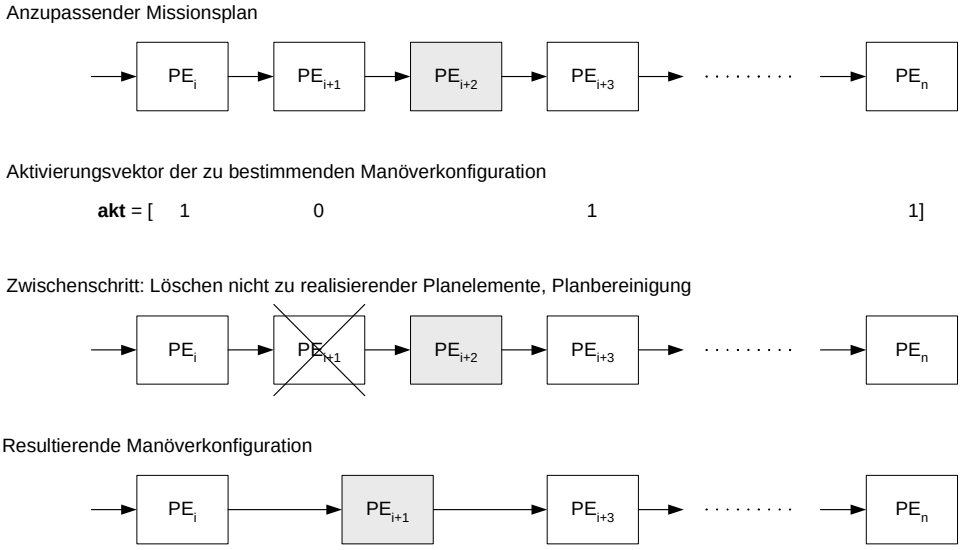


Abbildung 6.14: Erzeugen einer Manöverkonfiguration, die grau hinterlegten Planelemente stellen Füllplanelemente dar

Im ersten Schritt der Optimierung werden die Aktivierungsvektoren entsprechend dem zugehörigen Wert der Manöverkonfiguration absteigend sortiert (Abbildung 2). Beginnend mit der höchstwertigsten Konfiguration werden nun im zweiten Schritt der Reihe nach die Nebenbedingungen berechnet und geprüft. Die erste Konfiguration, die alle Bedingungen erfüllt, ist die gesuchte optimale Lösung. Sie besitzt den höchsten Wert unter den vorhandenen Beschränkungen.

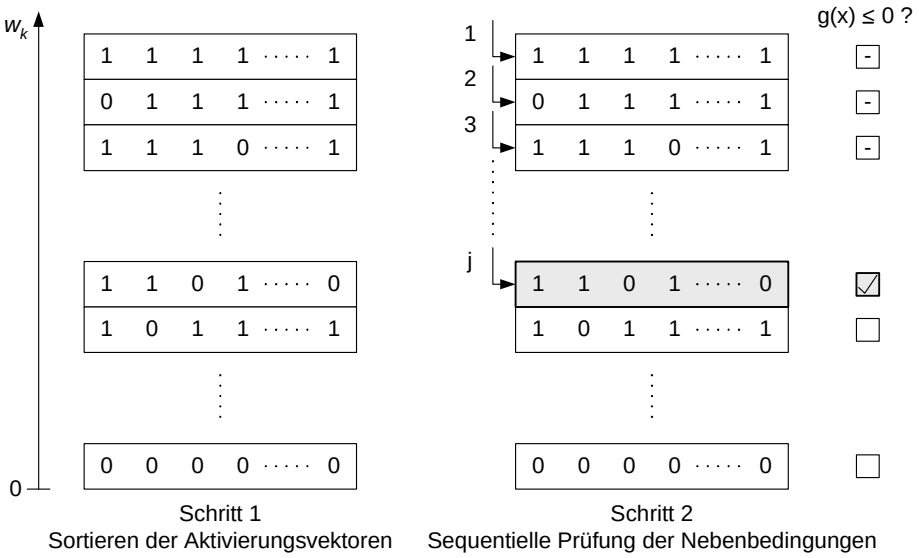


Abbildung 6.15: Zweistufiges Konzept der Optimierung

Heuristik zur schnellen Problemlösung

[Ree95] definiert eine Heuristik als Technik, die mit vertretbarem Rechenaufwand gute approximative Lösungen sucht. Sie ist jedoch nicht imstande, die Optimalität und Zulässigkeit des gefundenen Ergebnisses zu garantieren. Für das Rucksack-Problem existiert eine Vielzahl von Heuristiken, die bekannteste und einfachste ist der so genannte Greedy-Algorithmus[3]. Er besitzt einen geringen, linear mit der Problemgröße anwachsenden Zeitbedarf.

[3]greedy = gierig, gefräßig

Dieses Verfahren wird auch für die Planoptimierung im Fall einer großen Anzahl von Planelementen verwendet. Ausgangspunkt ist analog zum Rucksackproblem eine absteigend sortierte Liste relativer Wertigkeiten w_{rel}, die auf die Kosten bezogen werden:

$$w_{rel}(k) = \frac{w_k}{\|\mathbf{c}_k\|} \quad k = [i,\ldots,m].$$

In jedem Schritt des Greedy-Algorithmus wird nun aus der Liste der verfügbaren Elemente das mit dem höchsten Wert ausgewählt, die Erfüllung der Nebenbedingungen der Optimierung laut Gleichung ? geprüft und daraufhin entschieden, ob dieses Element in der Auswahlliste verbleibt oder nicht benutzt wird. Sobald kein weiteres Element mehr vorhanden ist, terminiert das Verfahren und die Lösung ist gefunden. Hier zeigt sich auch der Charakter dieser Heuristik, der im Namen zum Ausdruck kommt: In jedem Schritt wird das wertvollste verfügbare Element genommen, ohne die Gesamtsituation zu berücksichtigen. Der große Vorteil dieses Algorithmus liegt darin, *eine* Lösung in verhältnismäßig kurzer Zeit zu finden. Die Qualität des Ergebnisses und der Zeitbedarf des Verfahrens zeigen sich eindrucksvoll anhand der durchgeführten Untersuchungen für DeepC.

6.1.5 Chart Server - Geografische Planprüfung und –modifikation

Um eine sichere Navigation eines sich in seiner Umwelt bewegenden Roboters zu gewährleisten, sollte der Missionsplan vor dem Start der Mission und nach einer Umplanung auf Einhaltung der Beschränkungen durch seine Umgebung überprüft werden. Die zu berücksichtigenden Restriktionen können vielfältig sein. So muss ein in Gebäuden operierendes System eine Karte der Flure, Räume und Hindernisse besitzen, um sinnvolle Bewegungen durchführen zu können [Kni91]. Ein im Freien arbeitender Roboter benötigt hingegen ein Geländemodell und - falls er auf befestigte Wege angewiesen ist - die Informationen über verfügbare Fahrstrecken.

Die hier vorzustellenden Verfahren zur geografischen Planprüfung und -modifikation sind auf Systeme ausgerichtet, die sich in der natürlichen Umwelt "'fliegend"' bewegen. Dabei kann es sich sowohl um Flug- als auch Tauchroboter handeln. Sie sind dadurch gekennzeichnet, dass auf Verletzung der Umgebungsbeschränkungen mit dreidimensionalen Manövern reagiert werden kann. Dies spielt insbesondere bei der Planmodifikation eine große Rolle. Voraussetzung für die Anwendung der folgenden Algorithmen ist das Vorhandensein der geografischen Position des autonomen mobilen Systems im Zustandsvektor, was im Allgemeinen durch den Begriff *mobil* impliziert wird.

Damit die Position eines sich im Freien bewegenden Roboters festgelegt ist, muss ein eindeutiges Koordinatensystem verwendet werden. Für die Lagebeschreibung auf der Erdoberfläche bieten sich das geografische und das geozentrische Koordinatensystem an [HGM02]. Letzteres ist ein erdfestes dreidimensionales rechtwinkliges System (X,Y,Z), das seinen Ursprung im Erdmittelpunkt hat (Abbildung 6.16 (a)). Gebräuchlicher sind jedoch die geografischen Flächenkoordinaten geografische Breite λ und geografische Länge φ, die auch in dieser Arbeit benutzt werden. Die Flächenkoordinaten sind immer bezogen auf eine bestimmte Beschreibung der Erdfigur, meist wird der Geoid[4] durch eine Kugel oder einen Rotationsellipsiod angenähert (Abbildung 6.16 (b)). Aufgrund der weiten Verbreitung des Global Positioning System (GPS) ist mittlerweile der als WGS84 (World Geodetic System) bezeichnete Ellipsoid der am häufigsten verwendete.

Die Beschreibung mittels Flächenkoordinaten gestattet lediglich die Festlegung einer Position auf dem damit assoziierten Ellipsoid, erst die Hinzunahme einer Höhe über (bzw.

[4]Niveaufläche (Äquipotentialfläche) des Erdschwerefeldes, die in mittlerer Höhe des Meeresspiegels verläuft

Tiefe unter) diesem Referenzobjekt erlaubt eine eindeutige Bestimmung der Lage eines Roboters. Der verwendete Positionsvektor ergibt sich damit zu (*λ*, *φ, h)*.

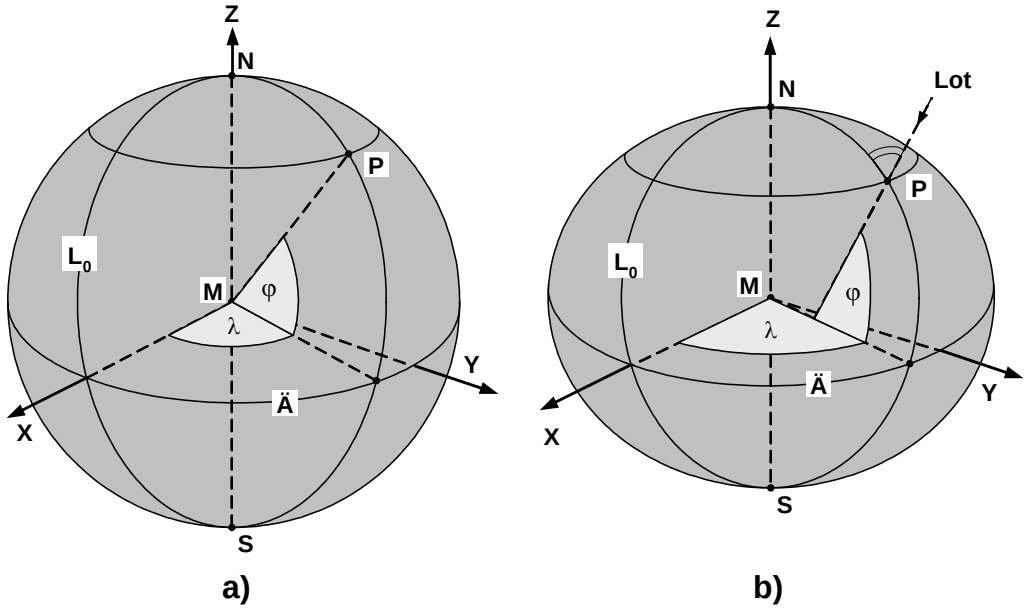


Abbildung 6.16: Geografisches Koordinatensystem (a) auf der Kugel und (b) auf dem Rotationsellipsiod (nach [?])

6.1.5.1 Datenbasis

Grundlage der im Folgenden vorzustellenden Verfahren ist eine auf diesen Koordinaten basierende digitale Karte des Einsatzgebietes des Roboters. In dieser Karte ist die Erdoberfläche als Höhenmodell hinterlegt. Bedingt durch die Ausrichtung auf nicht erdgebundene Fahrzeuge (Flug- und Tauchroboter) sind keine künstlichen Fahrwege in das Geländemodell integriert worden.

Als Datenstruktur für eine effiziente Prüfung des Missionsplanes werden dreiecksvermaschte Netze (engl. TIN = Triangulated Irregular Network) verwendet. Diese bestehen aus einer Menge ungleichmäßig verteilter Punkte, die durch Verbindungen untereinander Dreiecke formen. Der Vorteil eines TIN gegenüber einer regelmäßigen Verteilung der Punktdaten besteht darin, dass Bereiche mit unterschiedlicher Informationsdichte gleichzeitig dargestellt werden können. Die Erdoberfläche ist besonders dadurch gekennzeichnet, dass sich häufig großflächige Ebenen geringer Höhenänderungen mit vielfältig strukturierten Gegenden abwechseln. Dreiecksvermaschte Netze sind geradezu prädestiniert, derartige Formen detailgetreu abzubilden.

Jeder Geländepunkt im TIN ist charakterisiert durch seine geografischen Koordinaten (*λ*, *φ*) und besitzt die Höhenangabe *h* als Attribut. Die Dreiecke werden mit Hilfe der zweidimensionalen Delaunay Triangulation über *λ* und *φ* erzeugt [PS90]. Bei diesem Verfahren wird das so genannte Umkreiskriterium als Grundlage für die Dreiecksbildung verwendet. Es besagt, dass im Umkreis eines Dreiecks kein weiterer Punkt liegen darf (Abbildung 6.17). Durch diese Bedingung wird stets der kleinste Innenwinkel der Dreiecke maximiert.

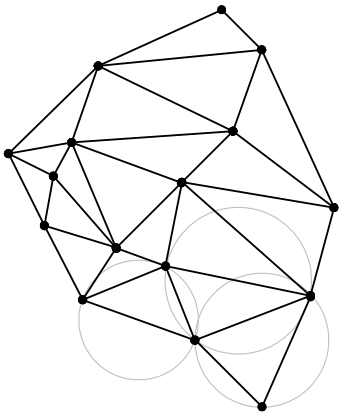


Abbildung 6.17: Delaunay Triangulation mit Umkreisen

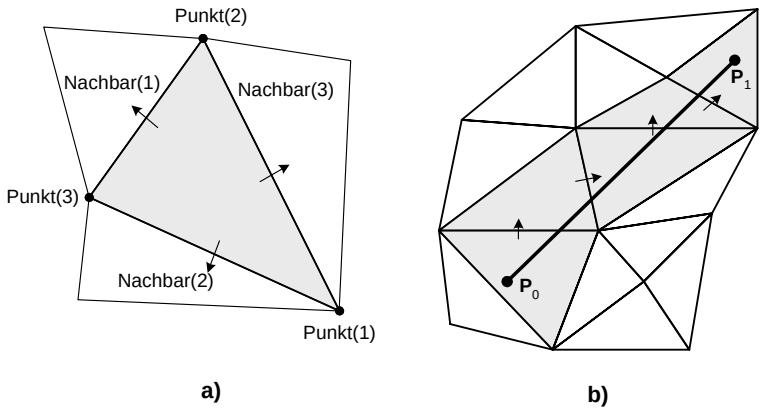


Abbildung 6.18: Datenstruktur eines Dreiecks (a) und Navigation innerhalb des dreiecksvermaschten Netzes (b)

Die interne Datenstruktur des dreiecksvermaschten Netzes ist auf eine effiziente Arbeitsweise optimiert [Dev98]. Jedes Dreieck besitzt Verweise auf die drei Eckpunkte sowie die benachbarten Dreiecke (Abbildung 6.18 (a)). Damit ist eine schnelle Navigation innerhalb der Datenbasis möglich, was besonders wichtig für die Prüfung eines Missionsplanes ist. So kann in kürzester Zeit ein den Punkt $\mathbf{P}_0=[\lambda, \varphi]$ umschließendes Dreieck gefunden und daraufhin sehr effektiv eine Liste der Dreiecke bestimmt werden, die die Strecke von $\mathbf{P}_0$ nach $\mathbf{P}_1$ berühren (Abbildung 6.18 (b)).

6.1.5.2 Prüfung des Missionsplanes

Ziel der Missionsplanprüfung ist es, eine Aussage über die Fahrbarkeit einer geplanten Mission zu treffen. Dazu ist eine Reihe von Operationen notwendig (Abbildung 6.19). Der Plan wird zunächst in seine Planelemente aufgespalten. Je nach Art eines Elementes können unterschiedliche geometrische Figuren als Beschreibungsform gefunden werden, die die Fahrkurve des mobilen Systems bei der Planrealisierung repräsentieren. Um die Prüfung der Fahrkurven gegen die Datenbasis effizient zu gestalten, werden nur Geraden (genauer gesagt Strecken) zur Schnittberechnung verwendet. Dafür existiert eine Vielzahl leistungsfähiger Verfahren. Alle weiteren geometrischen Objekte können – zum Beispiel über polygonale Approximation - durch Geradenstücke angenähert werden. Für eine effiziente Schnittberechnung zwischen Strecken und Dreiecken bietet sich die Transformation der Dreiecke in baryzentrische Koordinaten an, die im nachfolgenden Abschnitt näher erläutert wird.

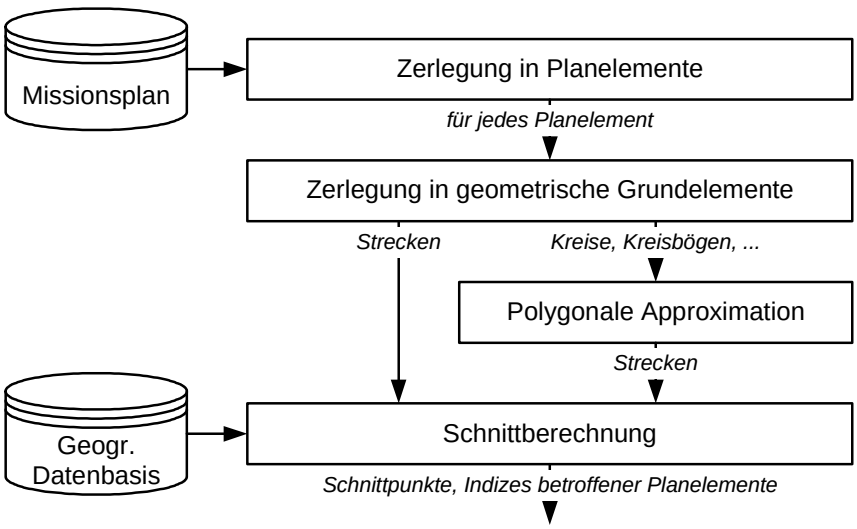


Abbildung 6.19: Ablauf der geografischen Prüfung eines Missionsplanes

Schnittberechnungen

Die Kollisionsprüfung des Missionsplanes kann mit Hilfe der vorgestellten Verfahren zur Planzerlegung auf Schnittberechnungen zwischen Strecken und Dreiecken zurückgeführt werden. Da sich Dreiecke mit üblichen x-y-Koordinaten schwer beschreiben lassen, werden so genannte baryzentrische Koordinaten (*u, v, w*) verwendet. Mit ihrer Hilfe kann ein beliebiger Punkt in der Dreiecksebene als Linearkombination der Eckpunkte $\mathbf{P}_0$, $\mathbf{P}_1$ und $\mathbf{P}_2$ dargestellt werden:

$$P(u,v,w) = wP0 + uP_1 + vP_2. \tag{6.1}$$

Liegt **P** innerhalb des Dreiecks, erhalten (*u, v, w*) eine physikalische Bedeutung. Sie entsprechen dem Flächeninhalt der Dreiecke $\mathbf{PP}_1\mathbf{P}_2$, $\mathbf{P}_0\mathbf{PP}_2$ und $\mathbf{P}_0\mathbf{P}_1\mathbf{P}$. Bei einer Zuweisung $u = v = w = 1/3$ befindet sich aufgrund der Flächengleichheit der Punkt **P** im Schwerpunkt des Dreiecks. Aus diesem Grund werden baryzentrische Koordinaten auch als Schwerpunktkoordinaten bezeichnet. Sie sind ein weit verbreitetes Beschreibungsmittel in der Finite-Element-Methode [GRT93] und transformieren jedes Dreieck in ein Referenzdreieck, wenn $w = 1 - u - v$ gesetzt wird (Abbildung 7). Diese Festlegung wird bei den Schnittberechnungen ebenfalls genutzt.

Baryzentrische Koordinaten sind von dem den Punkten $\mathbf{P}_i$ zugrunde liegenden Koordinatensystem unabhängig und besitzen deshalb günstige Eigenschaften für die Schnittberechnung. So kann anhand der Werte für (*u, v*) sofort entschieden werden, ob der Punkt **P** zur Dreiecksfläche $\mathbf{P}_0\mathbf{P}_1\mathbf{P}_2$ gehört oder nicht:

$$\mathbf{P}(u,v) \text{ gehört zum Dreieck} \leftrightarrow \begin{cases} u \geq 0 \\ v \geq 0 \\ u + v \leq 1. \end{cases} \tag{6.2}$$

In [MT97] wird ein Verfahren zur Schnittpunktberechnung zwischen Dreiecken und Geraden im dreidimensionalen Raum vorgestellt, dass diese Bedingungen verwendet. Darin wird eine Gerade durch die Punkte $\mathbf{G}_0$ und $\mathbf{G}_1$ mit Hilfe eines Parameters *t* folgendermaßen dargestellt:

$$\mathbf{G}(t) = \mathbf{G}_0 + t\left(\mathbf{G}_1 - \mathbf{G}_0\right) \qquad t \in \Re. \tag{6.3}$$

Schnittpunkte existieren genau dann, wenn die Lösung (*u', v', t'*) des Gleichungssystems $\mathbf{P}(u, v, 1 - u - v) = \mathbf{G}(t)$ den Zusammenhang (5) erfüllt. Da für die Planprüfung keine Geraden, sondern Strecken mit Start- und Endpunkt verwendet werden, muss dieser Algorithmus um die Bestimmung der Zugehörigkeit zu der Strecke $\overline{\mathbf{G}_0\mathbf{G}_1}$ ergänzt werden. Mit Hilfe des Parameters t ist die Entscheidung über die Zugehörigkeit zur Strecke einfach zu treffen:

$$\mathbf{G}(t) \text{ gehört zur Strecke} \leftrightarrow 0 \leq t \leq 1. \tag{6.4}$$

Mit diesem Ansatz steht eine sehr effiziente Methode zur Verfügung, um eine Planprüfung in kürzester Zeit durchführen zu können. Dabei wird für jede Strecke, die aus den Manövern des Missionsplanes gebildet werden kann, eine Liste der überdeckten Dreiecke entsprechend Abbildung 3 ermittelt. Schnittberechnungen zwischen den betroffenen Dreiecken und dem zugehörigen Streckenabschnitt liefern dann die Aussage, ob eine Verletzung der Beschränkungen durch das digitale Höhenmodell vorliegt oder nicht. Dieses Ergebnis wird anschließend in der geografischen Planmodifikation für die Auswahl der neu zu berechnenden Fahrstrecken genutzt.

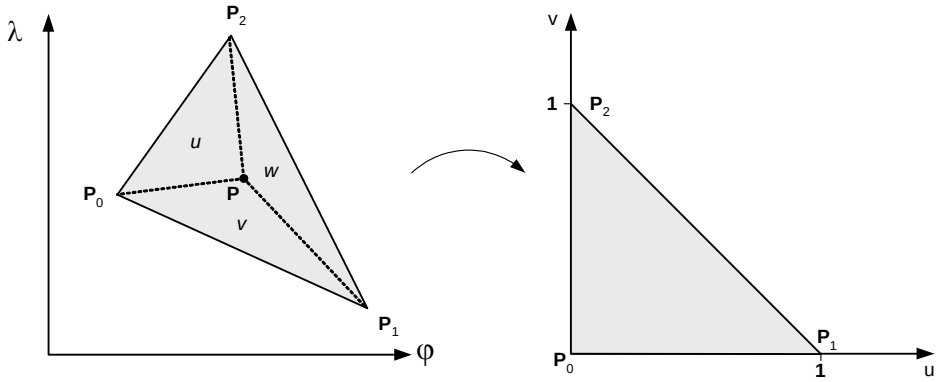


Abbildung 6.20: Baryzentrische Koordinaten im Dreieck

6.1.5.3 Modifikation des Missionsplanes

Wird bei der Prüfung des Missionsplanes festgestellt, dass Verletzungen der Beschränkungen durch das Terrain auftreten, generiert die geografische Planmodifikation einen Vorschlag für die Anpassung des oder der betroffenen Planelemente. Im Ergebnis entsteht dabei eine neue Route zwischen geografischer Start- und Endposition dieser Manöver.

Die im Abschnitt 6.1.5.1 präsentierte Struktur zur Hinterlegung eines digitalen Höhenmodells ist für die Berechnung eines neuen Weges nicht gut geeignet. Eine wesentliche Ursache dafür liegt in der fehlenden Information über die Fahrbarkeit der Strecken zwischen einzelnen Knotenpunkten des Netzes. Außer für Wege zwischen den drei Eckpunkten eines Dreiecks, die sich immer als direkte Verbindung ergeben, kann keine Route ohne größeren Rechenaufwand bestimmt werden. Darüber hinaus steigt der Aufwand mit zunehmender Zahl der Knoten des Netzes. Aus diesem Grund ist eine Vorverarbeitung der Daten und damit eine Erweiterung der Datenbasis unumgänglich.

Als geeignete Verfahren für die Suche des nach Möglichkeit minimalen Weges zwischen zwei Punkten im Gelände bieten sich graphenbasierte Methoden an, die im schlechtesten Fall polynomial von der Größe des Graphen abhängen [BG93]. Sie garantieren eine effiziente Modifikation des Missionsplanes und werden im Abschnitt *Graphenbasierte Routensuche* vorgestellt.

Erweiterung der Datenbasis

Aufgabe der Vorverarbeitung ist die Generierung der für die dreidimensionale Wegplanung benötigten Informationen und deren Integration in die Datenbasis. Für graphenbasierte Verfahren wird ein Graph $G = (V,A)$ benötigt, der als Knoten V die Stützstellen des TIN und als Kanten A die Kosten für den Weg zwischen den Knoten beinhaltet. Der Graph kann dabei sowohl gerichtet als auch ungerichtet sein. Ein gerichteter Graph verknüpft die Kosten für den zurückzulegenden Wegabschnitt mit der Bewegungsrichtung, so dass der Aufwand für die Fahrt von Knoten *a* nach *b* anders als der von *b* nach *a* sein kann. Ein ungerichteter Graph besitzt demgegenüber Kantenbewertungen, die unabhängig von der Richtung der Bewegung sind. Im Hinblick auf den Einsatz der Verfahren zur geografischen Planmodifikation für das autonome Unterwasserfahrzeug DeepC wird hier ein ungerichteter Graph verwendet (Abbildung 6.21). Die Ursache dafür liegt in dem ähnlichen Aufwand, der für Tiefenänderungen in positiver und negativer Richtung notwendig ist. Durch das Vorhandensein eines statisch wirkenden Trimmsystems ist im Gegensatz zu fliegenden Systemen für beide Fälle ein höherer Energiebedarf als für die Bewegung in der Horizontalen vorhanden.

Die Kosten $c_{i,j}$ für die Fahrstrecke vom Knoten i zum Knoten j ergeben sich über eine Bewertungsfunktion, die im Fall eines mobilen Roboters immer positive Werte liefert

(entsprechend dem Ressourcenverbrauch beim Zurücklegen der Strecke). Nicht fahrbare Strecken besitzen ungültige Kostenangaben (z.B. ∞). Dieser Graph wird in Form einer Distanzmatrix in die Datenbasis aufgenommen (Abbildung .6.22). Die Kosten unterhalb der Diagonalen entsprechen bei einem ungerichteten Graphen wie dem hier dargestellten denen im oberen Dreieck des Distanzmatrix: $c_{i,j} = c_{j,i}$.

Für nicht erdgebundene Fahrzeuge, die sich frei im dreidimensionalen Raum bewegen können, ist eine einmalige, vollständige Bestimmung des Graphen nicht möglich. Die Ursache dafür liegt in dem Sachverhalt, dass die Fahrbarkeit zwischen zwei dreidimensionalen Punkten im Gelände von deren Abstand zum Boden abhängig ist. Ein in 10 Kilometer Höhe fliegender Roboter benötigt zum Beispiel außer bei Start und Landung keine Informationen über die Struktur des Terrains, da bekanntlich diese Höhe an keinem Ort der Erdoberfläche erreicht werden kann. Soll er jedoch eine Strecke fliegen, deren Start- und Endposition im Abstand von 250 Meter zum Boden liegen, sind vielfältige Beschränkungen innerhalb der verfügbaren Routen (und damit auch im Graphen) zu erwarten. Daraus ergibt sich prinzipiell die Notwendigkeit, den Graphen anhand der Höhenangaben im Missionsplan immer wieder neu zu erzeugen.

Die Ermittlung des Graphen erfordert jedoch durch Berechnung der Fahrbarkeit (oder auch Sichtbarkeit) zwischen den einzelnen Knoten des TIN einen großen Rechenaufwand. Deshalb kann häufig nicht während einer Mission eine komplette Neugenerierung des Graphen durchgeführt werden. Als Ausweg wird eine Strategie vorgeschlagen, die den Gesamtprozess in eine offline und eine online Phase aufteilt:

- *offline Phase:* Es erfolgt die Fahrbarkeitsprüfung zwischen allen Paaren von Knoten des Graphen mit minimal zulässigem Abstand zum Gelände. Dies ergibt eine geringe Anzahl von fahrbaren Strecken, die aber in jeder beliebigen Höhe genutzt werden dürfen. Die Berechnungen sind sehr zeitaufwendig, da für jedes Knotenpaar Schnittberechnungen mit dem digitalen Höhenmodell notwendig sind. Die Kosten werden über den zurückzulegenden dreidimensionalen Weg ermittelt.
- *online Phase:* Die Höheninformationen von Anfangs- und Endpunkt einer zu suchenden Route werden in den Graphen integriert. Dabei werden die in der offline Phase gefundenen Routen daraufhin überprüft, ob der Roboter bei ihrer Benutzung Höhenänderungen durchzuführen hat. Auftretende Änderungen werden mit zusätzlichen Kosten belegt. Der Rechenaufwand für diese Schritte ist wesentlich geringer als der der offline Phase, da nur eine geringe Anzahl von Knotenpaaren untersucht werden muss. Bei den in Experimenten untersuchten Daten ergab sich, dass ca. 10 bis 20 % der Gesamtzahl der Knotenpaare die offline durchgeführte Fahrbarkeitsprüfung bestanden haben und damit in den Graphen aufgenommen wurden.

  Zusätzlich werden Start- und Endpunkt der Route in den Graphen integriert, wenn sie nicht bereits als Knoten vorhanden oder in geringem, parametrierbaren Abstand zu einem vorhandenen Knoten gelegen sind.

Durch die offline Vorbereitung des Graphen ist die zur Verfügung stehende Routenzahl gegenüber der theoretisch möglichen eingeschränkt, da nur die in allen Höhen befahrbaren Verbindungen zwischen den Knoten ermittelt werden. So wird nicht unbedingt der optimale Weg durch das Gelände gefunden, sondern die mit der vorhandenen (vorverarbeiteten) Datenbasis kürzeste Route. Demgegenüber steht als großer Vorteil, dass diese Verfahrensweise auf den oft begrenzten Rechenkapazitäten eines mobilen Roboters eingesetzt werden kann.

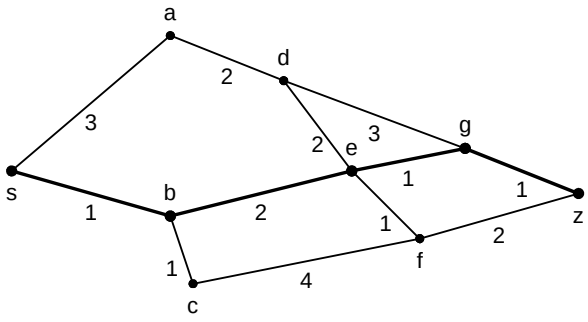


Abbildung 6.21: Graph mit positiven Kantenbewertungen

	s	a	b	c	d	e	f	g	z
s	0	3	1	∞	∞	∞	∞	∞	∞
a		0	∞	∞	2	∞	∞	∞	∞
b			0	1	∞	2	∞	∞	∞
c				0	∞	∞	4	∞	∞
d					0	2	∞	3	∞
e						0	1	1	∞
f							0	∞	2
g								0	1
z									0

Abbildung .6.22: Distanzmatrix zum Graph aus Abbildung 6.21

Graphenbasierte Routensuche

Mit Hilfe von Suchmethoden kann die kürzeste Route in einem Graphen G sehr effektiv ermittelt werden. Als Standardverfahren für ein derartiges Problem hat sich im Fall eines vollständig bekannten Graphen der nach seinem Erfinder E. W. Dijkstra benannte Dijkstra-Algorithmus etabliert [Dij59]. Er findet in polynomialer Zeit iterativ eine Route, falls diese in dem Graphen existiert.

Ausgehend von einem Startknoten *s* werden beim Dijkstra-Algorithmus alle benachbarten Knoten (Knoten, die von s aus erreichbar sind, also eine Kantenbewertung ungleich ∞ besitzen) besucht und die jeweiligen Kosten bestimmt (Abbildung 6.23). Referenzknoten k_{ref} für die zweite Iteration wird immer derjenige, der die niedrigsten Kosten zum Startknoten besitzt (Knoten *b*). Es erfolgt wiederum eine Kostenermittlung zu den Nachbarknoten von k_{ref}, die zu den bereits aufgelaufenen, auf *s* bezogenen Kosten hinzugefügt werden. Iterativ werden nun nacheinander die Knoten mit den jeweils niedrigsten Gesamtkosten als Referenzknoten festgelegt, wobei bereits besuchte als erledigt markiert und nicht nochmals ausgewählt werden.

	Knoten									*erledigt*
	s	a	b	c	d	e	f	g	z	
Start	0	∞	∞	∞	∞	∞	∞	∞	∞	-
Iteration 1	0	3	1	∞	∞	∞	∞	∞	∞	s
Iteration 2	0	3	1	2	∞	3	∞	∞	∞	s,b
Iteration 3	0	3	1	2	∞	3	6	∞	∞	s,b,c
Iteration 4	0	3	1	2	5	3	6	∞	∞	s,b,c,a
Iteration 5	0	3	1	2	5	3	4	4	∞	s,b,c,a,e
Iteration 6	0	3	1	2	5	3	4	4	6	s,b,c,a,e,f
Iteration 7	0	3	1	2	5	3	4	4	5	s,b,c,a,e,f,g
Iteration 8	0	3	1	2	5	3	4	4	5	s,b,c,a,e,f,g,d

Abbildung 6.23: Ablauf des Dijkstra-Algorithmus für den Graph aus Abbildung 6.21

Im Ergebnis entsteht ein Kostenvektor, der den Aufwand für alle Strecken ausgehend von *s* repräsentiert. Gleichzeitig ergibt sich eine Liste derjenigen Knoten, die den kürzesten Weg vom Start zum Ziel bilden. Mit Hilfe ihrer Positionsangaben wird beispielsweise das in Abbildung 6.24 dargestellte, nicht fahrbare Manöver PE_{i+1} des Missionsplanes durch eine Sequenz von Planelementen (PE_{G1},...,PE_{Gn}) ersetzt.

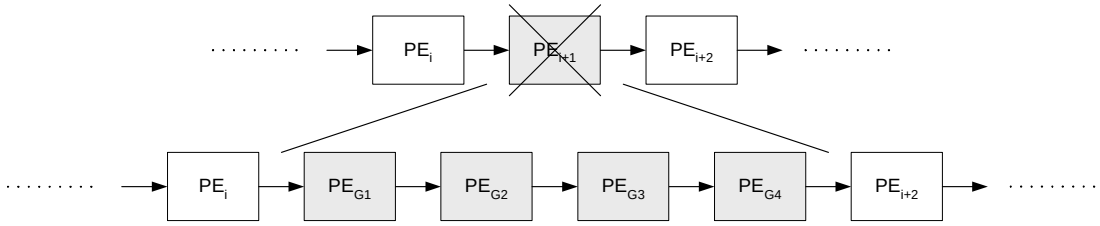


Abbildung 6.24: Geografische Planmodifikation

6.1.6 Anwendung für DeepC

6.1.6.1 Softwarearchitektur

DeepC besitzt eine Softwarearchitektur, die weitestgehend einem menschlichen Handlungsmodell nach Rasmussen entspricht [Ras83]. Nachfolgend werden die drei Ebenen des Modells vorgestellt. Dabei wird insbesondere die für das intelligente Missionsmanagement wesentlichen Module und ihre Verknüpfungen eingegangen (Abbildung 6.25).

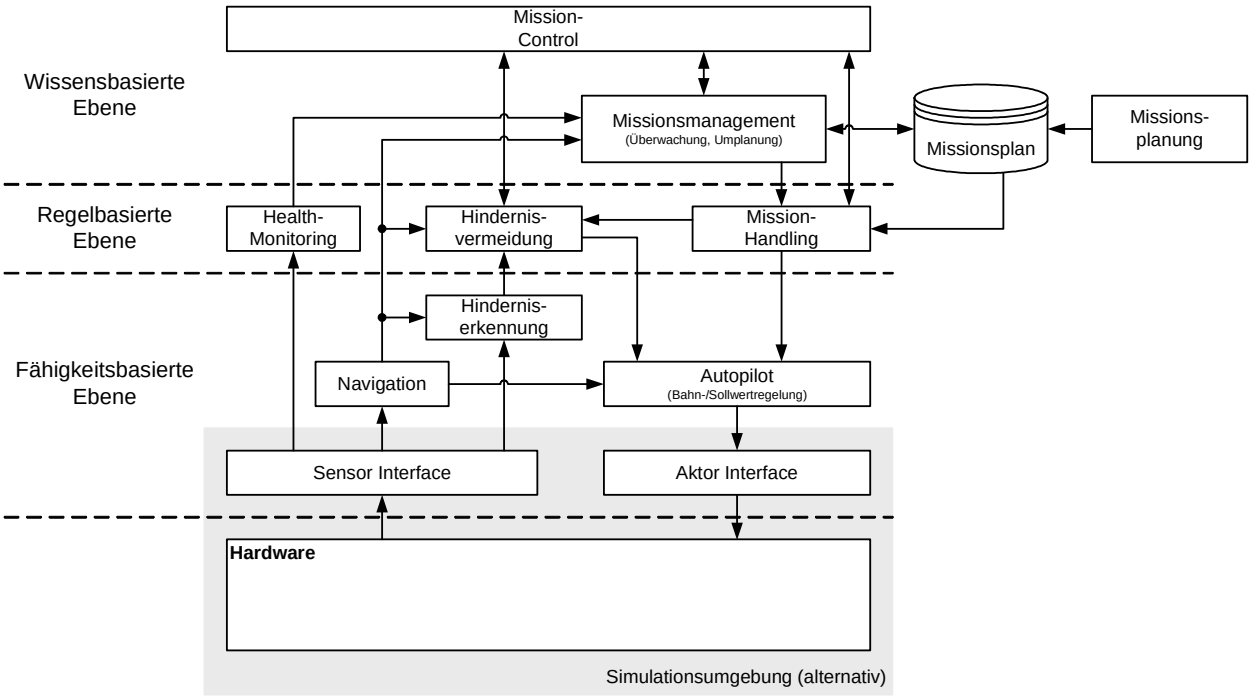


Abbildung 6.25: Softwarearchitektur des AUV DeepC (Auszug)

Die *fähigkeitsbasierte Ebene* beinhaltet die grundlegenden Module für die Fahrzeugsteuerung. Dazu zählen neben dem Autopiloten auch die Module zur Aufbereitung und Fusion der von den verschiedenen Sensoren gelieferten Informationen. Die Schnittstellen zur Hardware des Fahrzeuges werden über Interface-Module realisiert. Anstelle der Hardware kann auch eine Simulationsumgebung integriert werden, die im Vorfeld der praktischen Erprobung eine weitreichende Untersuchung der erstellten Softwaremodule erlaubt. Die Navigation verknüpft die Informationen der unterschiedlichen Sensoren und generiert daraus den Navigationsdatensatz. Dieser Datensatz ist eine wesentliche Grundlage für die Arbeit der anderen Softwareeinheiten. Die vom Sonar gelieferten Messwerte werden durch die Hinderniserkennung ausgewertet, um Position, Größe und Geschwindigkeit von potenziellen Hindernissen zu ermitteln. Dafür sind entsprechende Bildverarbeitungsmethoden entwickelt worden.

Das Mission-Handling teilt auf der *regelbasierten Ebene* die komplexen Manöver in zugehörige Basismanöver auf, die an den Autopiloten übergeben werden. Als Basismanöver werden die für die komplexen Planelemente notwendigen geometrischen Grundkonstrukte (z.B. Gerade, Kreis, Kreisbogen) bezeichnet. Darüber hinaus beinhaltet das Mission-Handling wichtige Verwaltungsfunktionen für den Missionsplan. Die Hindernisvermeidung bewertet die von der Hinderniserkennung erfassten Objekte im Umfeld des Fahrzeuges hinsichtlich möglicher Kollisionen. Dabei werden auch die nächsten zu realisierenden Basismanöver einbezogen. Das Health-Monitoring führt eine Diagnose der Hardware des Fahrzeuges durch und erzeugt in verschiedenen Situationen Vorschläge für die

durchzuführenden Handlungen bei Auftreten defekter Subsysteme im AUV. Diese Vorschläge und die Ergebnisse der Diagnose fließen in das intelligente Missionsmanagement ein.

Die *wissensbasierte Ebene* beinhaltet das intelligente Missionsmanagement, das in Abschnitt 2 näher vorgestellt wird. Als zentrale Entscheidungsinstanz des gesamten Systems ist das Modul Mission-Control für die Koordinierung der Arbeit der verschiedenen Programme verantwortlich. So muss zum Beispiel vor der Durchführung von Modifikationen des Missionsplanes eine entsprechende Genehmigung von Mission-Control vorliegen. Darüber hinaus überwacht dieses Modul alle ablaufenden Prozesse, um frühzeitig ein Fehlverhalten einzelner Einheiten des AUV zu erkennen.

6.1.6.2 Manöverkatalog

Der Manöverkatalog beinhaltet die Menge verfügbarer komplexer Manöver für DeepC. Er wurde von den Projektpartnern im Laufe des Projektes festgelegt und von ATLAS Elektronik in entsprechender Software implementiert. Der Katalog umfasst die folgenden Manöver, die für das Missionsmanagement eine zentrale Rolle spielen:

- *TRACK,*
- *MEANDER,*
- *Q-ROUTE,*
- *GPS-UPDATE,*
- *SURFACE (AUFTAUCHEN),*
- *DESCENT (ABTAUCHEN),*
- *SV_PROFILE (Sound Velocity Profile),*
- *CIRCLING.*

Verschiedene Manöver des Manöverkatalogs können als so genannte *Event-Manöver* bei der Missionsplanung in ihrer Geometrie beschrieben und im späteren Einsatz bei dem Missionsmanagement angefordert werden. Damit kann bei Auftreten bestimmter, vor dem Missionsstart bekannter Situationen mit vordefinierten Aktivitäten des AUV reagiert werden. Ein Beispiel dafür ist die Durchführung einer Kabelinspektion, bei der bei einem Verlust des zu inspizierenden Kabels durch die entsprechende Sensorik ein Suchmanöver an der letzten bekannten Position mit gültigem Signal ausgeführt wird. Der zu nutzende Manövertyp und die Ausdehnung des Suchgebietes werden durch den Operator in Form eines Event-Manövers vor dem Missionsstart festgelegt, Positionierung und Ausrichtung erfolgen dann anhand der vorhandenen Navigationsdaten durch das intelligente Missionsmanagement.

Diese komplexen Manöver werden von dem Modul Mission-Handling in eine Folge von Basismanövern zerlegt. Der Autopilot realisiert die einzelnen Basismanöver, überwacht das Erreichen ihres Endpunktes und fordert daraufhin das nachfolgende Element an.

6.1.6.3 Bestandteile und Arbeitsweise des Einsatzführungssystems

6.1.6.3.1 Missionsüberwachung

Die Missionsüberwachung wertet Informationen unterschiedlicher Herkunft aus, um eine Entscheidung zum Umplanen der Mission zu treffen. Als wesentlichste Informationsquellen dienen dabei der Navigationsdatensatz und die Diagnoseergebnisse des Health-Monitoring. Diese Daten werden an das integrierte Expertensystem der Missionsüberwachung übergeben und während des nachfolgenden Prozesses der logischen Inferenz zur Aktivierung der betroffenen Regeln genutzt. In der Tabelle 3 ist eine Übersicht einiger durch die Missionsüberwachung zu bewältigender Umplanungssituationen dargestellt.

Den Schwerpunkt bilden dabei Reaktionen auf Defekte der Fahrzeugsensorik. Ziel ist es, möglichst wenig Einbußen an den geplanten Missionszielen bei gleichzeitiger minimaler

Gefährdung des AUV hinnehmen zu müssen. So wird zum Beispiel bei einem Defekt des inertialen Navigationssystems die Missionsausführung erst nach dem Ansteigen des geschätzten Navigationsfehlers über das definierte zulässige Maß abgebrochen. Die Positionsschätzung erfolgt in einer derartigen Situation mit Hilfe einer einfacheren Sensorik zur Lage- und Beschleunigungsmessung, die eine Weiterfahrt des Fahrzeuges ermöglicht.

Die Überwachung der vorhandenen Ressourcen beschränkt sich für DeepC auf die Betrachtung der elektrischen Energie, da die zeitlichen Aspekte einer Mission bereits durch das Modul Mission-Control beobachtet werden. werden. Ein ständig durchgeführter Vergleich der vorhandenen Energiereserven mit den vom Missionsplan benötigten Energiemengen soll frühzeitig Defizite aufdecken, die mittels der Planoptimierung zu beheben sind. Daneben ist auch die aktuelle Leistungsaufnahme des Fahrzeuges zu kontrollieren, da ein langfristiger, zu hoher Strombedarf das Energieerzeugungssystem unzulässig belasten würde. Diese Überwachung erfolgt durch das Health-Monitoring und führt gegebenenfalls zur Anforderung einer Geschwindigkeitsreduktion. Das intelligente Missionsmanagement muss daraufhin die Geschwindigkeiten des aktuellen sowie eventuell der nachfolgenden Manöver anpassen. Vom Health-Monitoring werden dazu Geschwindigkeit und Dauer der Reduktion berechnet und mit der Anforderung übermittelt.

Durch das Health-Monitoring können darüber hinaus auch ein sofortiger Missionsabbruch sowie eine Auftauchempfehlung an das Missionsmanagement übergeben werden. Beide Entscheidungen resultieren aus der Diagnose der Fahrzeugsysteme. Die Auswahl der durchzuführenden Handlung erfolgt durch das Diagnosemodul abhängig von der Schwere der vorliegenden Störung.

Das Nutzlastmodul ist ebenfalls in der Lage, einen begrenzten Einfluss auf den Missionsplan zu nehmen. Dazu sind während der Missionsplanung Event-Manöver mit ihrer Geometrie zu definieren, die durch das Missionsmanagement bei Bedarf parametriert und in den Plan aufgenommen werden. Während der Missionsdurchführung kann die Nutzlast nun in unterschiedlichen Situationen ein Event-Manöver anfordern. Ein Beispiel für ein derartiges Szenario ist die Durchführung einer Meeresbodenvermessung für das Verlegen von Seekabeln. Die dazu notwendige Sensorik erstellt laufend das zu den eingehenden Daten gehörige Profil des Bodens und kann damit auch auf eine Überschreitung definierter Grenzwerte für die Kabelverlegung (z.B. Neigungswinkel des Meeresbodens) reagieren. In einer derartigen Situation fordert die Nutzlast ein lokal begrenztes Manöver an der aktuellen Position an. Aus den zusätzlich gewonnen Daten kann bei der Planung des Kabelverlaufs eine Umgehung des betroffenen Geländeabschnittes eingearbeitet werden.

Abweichungen vom Missionsplan[5] erfordern unter Umständen ebenfalls Modifikationen des Planes. Exemplarisch sind in Tabelle 6.4 die notwendigen Korrekturen für einen zu großen Navigationsfehler sowie eine vertikale Abweichung von der geplanten Fahrstrecke aufgeführt.

Einige der in der Tabelle aufgeführten Umplanungsaufgaben werden durch die Missionsüberwachung mit Hilfe der Umplanungsbefehle bereits vollständig aufbereitet an die Missionsumplanung übergeben. Deren Tätigkeit besteht dann im Ausführen der einzelnen Umplanungsbefehle und in der Prüfung des entstehenden Missionsplanes. In den anderen Fällen sind durch die Umplanung zunächst die durchzuführenden Modifikationen zu ermitteln, die anschließend realisiert werden.

Einen Teil der zu einer Umplanung führenden Datensätze empfängt das intelligente Missionsmanagement zyklisch. Damit eine bereits in Bearbeitung befindliche Umplanung nicht nochmals ausgelöst wird, muss der aktuelle Zustand der betroffenen Regeln des

[5] z.B. infolge ungenauer Navigation oder schlechter Regelgüte (bedingt durch Strömung)

Expertensystems gespeichert werden. Das erfolgt über den einzelnen Regeln zugeordnete Fakten, die entsprechend Abschnitt 6.1.3.2.4 als Zustandspeicher verwendet werden.

Tabelle 6.4: Umplanungssituationen für DeepC (Auszug)

Situation	Umplanungsaufgaben
Defekte der Sensorik	
Defekt der Nutzlast	Entfernung aller Manöver aus dem Plan, die die Nutzlast verwenden
Defekt des INS	Abbruch der Mission, wenn der Navigationsfehler zu groß wird (mit Überprüfung der Erreichbarkeit einer Recovery-Position)
Defekt des DOLOG	Sofortiger Abbruch der Mission mit Überprüfung der Erreichbarkeit einer Recovery-Position
Akustik-Modem ist defekt	Setzen der Tiefe eines abschließenden Circling-Manövers im Plan auf 0.0
Ressourcenbezogene Umplanungen	
Mangel an Ressourcen (Energie)	Optimierung des Missionsplanes bei Unterschreiten der benötigten Ressourcen
Zu hohe Leistungsaufnahme	Anpassung der Fahrgeschwindigkeit der Manöver für eine begrenzte Zeit
Anforderungen durch das Health-Monitoring	
Unbedingter Missionsabbruch	Abbruch der Mission an der aktuellen Position, Einplanung der für den Abbruch benötigten Manöver
Auftauchempfehlung	Abbruch der Mission mit Überprüfung der Erreichbarkeit einer Recovery-Position
Anforderungen durch weitere Module	
Anforderung eines Event-Manövers	Einfügen des gewünschten Manövers in das aktuell ausgeführte Manöver
Detektierte Abweichungen vom Missionsplan	
Navigationsfehler zu groß	Einfügen eines GPS-Updates in das aktuelle Manöver
Zu große vertikale Trackablage	Einfügen eines Auftauch- / Abtauchmanövers von der aktuellen Tiefe auf die Endtiefe des aktuellen Manövers

6.1.6.3.2 Missionsumplanung

Die Missionsumplanung ist für die Ausführung der notwendigen Veränderungen am Missionsplan und deren Prüfung verantwortlich. Der Aufbau entspricht prinzipiell der in Abschnitt 6.1.2 vorgestellten Konzeption. Die interne Arbeitsweise unterscheidet sich im Wesentlichen durch die Einbindung des Missionsmanagements in die Softwarearchitektur des AUV, da alle durchzuführenden Aktivitäten einer Freigabe durch die zentrale Entscheidungsinstanz Mission-Control bedürfen. Zusätzlich ist der modifizierte Plan einer Kontrolle zu unterziehen, die sich aus der geografischen Planprüfung und -modifikation sowie einem Plausibilitäts-Check durch das Mission-Handling zusammensetzt. Erst wenn diese Prüfungen positiv abgeschlossen wurden, kann der neue Plan dem System zur Realisierung übergeben werden (Abbildung 6.26).

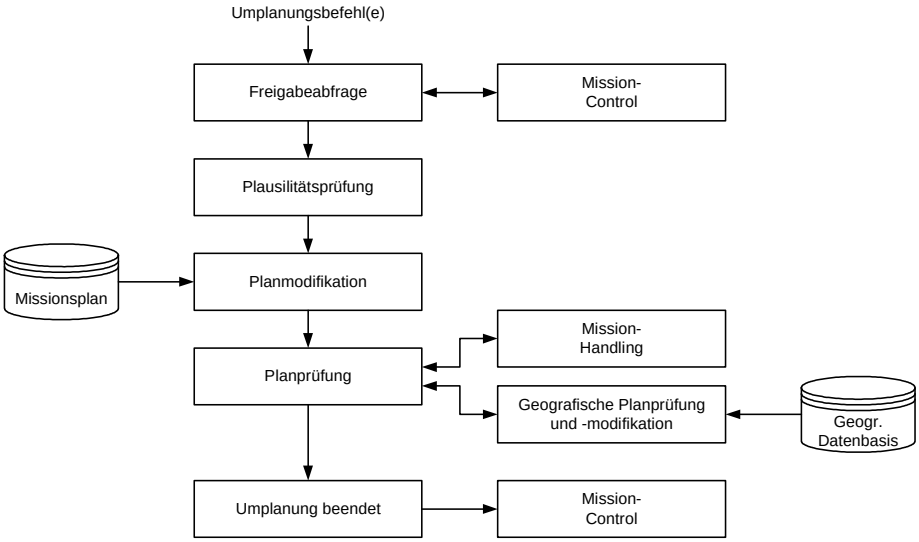


Abbildung 6.26: Ablauf der Missionsumplanung

Die geografische Planprüfung erfordert entsprechend den Aussagen aus Abschnitt 6.1.5 eine digitale Karte des Einsatzgebietes des autonomen mobilen Systems. Diese Karte muss aus den verfügbaren Informationen generiert und in der von dem intelligenten Missionsmanagement benötigten Struktur abgelegt werden.

Das im Forschungsprojekt vorgesehene erste Einsatzgebiet befindet sich in der Ostsee. Als Datenquelle für dieses Gebiet stehen digitale Höhenmodelle, häufig mit äquidistant angeordneten Stützstellen und digitale Navigationskarten für die Schifffahrt (Electronic Navigational Chart, ENC) zur Verfügung. Letztere beinhalten genaue Daten für die Navigation von Überwasserschiffen und besitzen ein standardisiertes Austauschformat [Int00]. In das zugrunde liegende objektorientierte Datenmodell sind alle für die Schifffahrt relevanten Informationen integriert. Die Datenbasis ist in Form von Zellen organisiert, die die gesamte Erdoberfläche in unterschiedlichen Genauigkeitsstufen abdecken. Jede Zelle enthält dabei die sie beschreibenden Objekte mit ihrer Geometrie und den zugehörigen Attributen. Abbildung 6.27 zeigt einen Ausschnitt aus einer Seekarte, der aus Gründen der Übersichtlichkeit lediglich die grundlegenden topografischen Elemente beinhaltet.

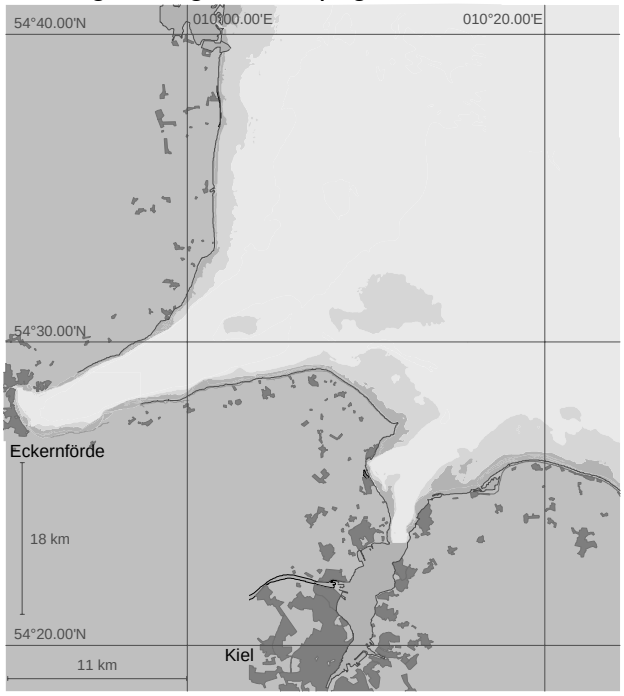


Abbildung 6.27: Digitale Seekarte der Eckernförder Bucht

Für das vorgesehene Einsatzgebiet von DeepC sind die Daten für Bereiche geringer Informationsdichte aus einem digitalen Höhenmodell [STK01] entnommen und mit der Seekarte aus Abbildung 6 kombiniert worden. Die abschließende Triangulation beinhaltet gleichzeitig eine Datenreduktion, um die Anzahl der äquidistant angeordneten Stützstellen des verwendeten Höhenmodells an die jeweiligen Bodenverhältnisse anzupassen. Dabei werden über den Gradient des Meeresbodenprofils diejenigen Messpunkte entfernt, die keinen maßgeblichen Beitrag zur Abbildung des Geländes leisten [SMK95]. Als Resultat liegt der benötigte Bereich der Ostsee als trianguliertes Höhenmodell vor (Abbildung 6.27).

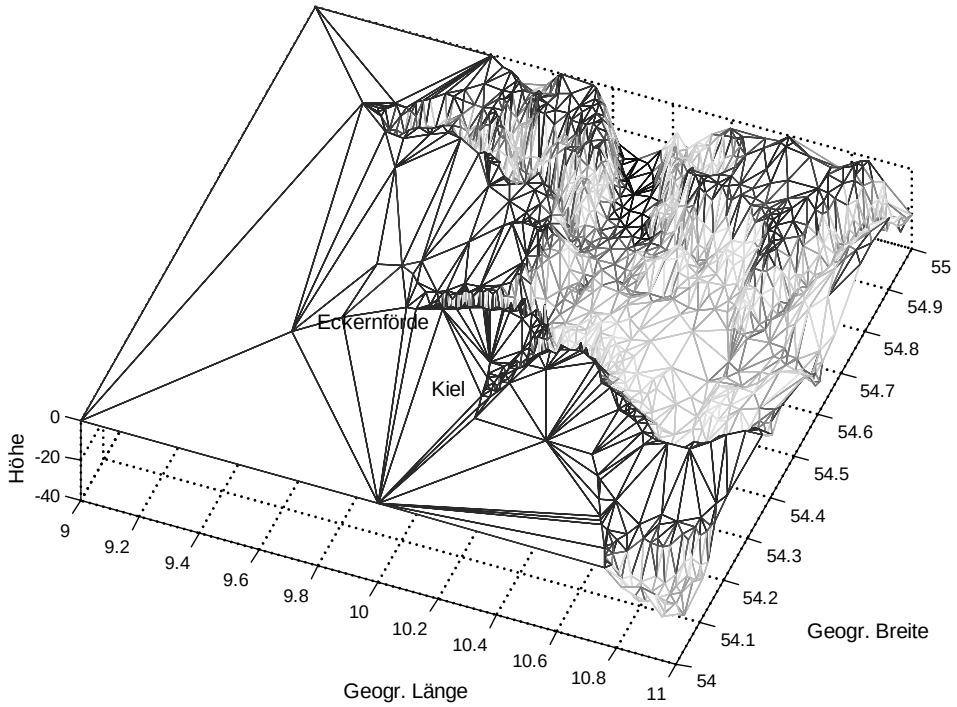


Abbildung 6.28: Höhenmodell für die geografischen Umplanungsverfahren

Die geografische Planmodifikation benötigt zusätzlich die Sichtbarkeits- oder auch Distanzmatrix der Höhenpunkte. Sie enthält Kostenbewertungen für die Strecken zwischen jeweils zwei Höhenpunkten, die bei einer Planprüfung als gültige Wege erkannt werden. Die Generierung des Graphen erfolgt entsprechend den Aussagen aus Abschnitt 6.1.5.3. In Tabelle 6.5 sind die wesentlichen Kenngrößen der resultierenden geografischen Datenbasis des intelligenten Missionsmanagements aufgeführt.

Tabelle 6.5: Daten des digitalen Höhenmodells

Kenngröße	Wert
West-Ost-Ausdehung	09°00,00'E bis 11°00,00'E (ca. 130 km)
Süd-Nord-Ausdehung	54°00,00'N bis 55°00,00'N (ca. 110 km)
Wassertiefe	0 - 33 m
Anzahl der Stützstellen	1775 = *n*
Anzahl der fahrbaren Wegstrecken	230793 = 14,6 % der maximal möglichen Strecken $\frac{1}{2} \cdot n \cdot (n-1) = 1574425$

Bei der Vorstellung des Manöverkataloges wurde bereits erwähnt, dass bei der Planbereinigung bedingt durch die freie Bewegung im dreidimensionalen Raum und die Unterscheidung von drei unterschiedlichen Transitmanövern Erweiterungen der in Abschnitt 6.1.4.2.5 vorgestellten Vorgehensweise notwendig sind. Im Einzelnen sind folgende Einschränkungen bei der Bereinigung eines Missionsplanes zu berücksichtigen:

- Manöver des Typs Gerade dürfen einen maximalen Neigungswinkel nicht überschreiten. Diese Begrenzung ergibt sich aus dem zulässigen Nickwinkel des AUV.

- Auf- und Abtauchmanöver gleichen nur Höhenunterschiede zwischen Anfangs- und Endposition aus, die Angaben zu Längen- und Breitengraden müssen stets identisch sein.

Die Planbereinigung selbst läuft als iterativer Prozess ab und fasst die folgenden Kombinationen von Transitmanövern zusammen (Abbildung 6.29):

a) Zwei oder mehrere Transitmanöver des Typs Gerade werden zu einem Manöver gleichen Typs kombiniert.
b) Ein zwischen zwei horizontalen Transitmanövern angeordnetes Auftauchmanöver wird vor dem kombinierten Manöver des Typs Gerade ausgeführt, um das AUV aus Sicherheitsgründen mit größerem Abstand zum Meeresboden fahren zu lassen.
c) Analog zur vorangehenden Verfahrensweise erfolgt ein Abtauchen erst nach der Durchführung des horizontalen Transits.
d) und e) Folgen vertikaler Transitmanöver werden abhängig von dem zu überwindenden resultierenden Höhenunterschied als einzelnes Abtauch- oder Auftauchmanöver ausgeführt.

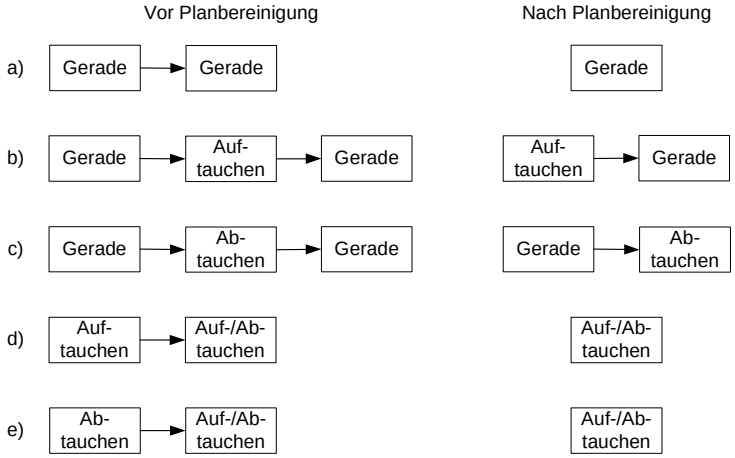


Abbildung 6.29: Vorgehensweise bei der Planbereinigung

Damit ist es möglich, vertikale Transitmanöver vollständig aus dem Plan zu entfernen. So wird bei der Planbereinigung geprüft, ob ein Höhenunterschied mit Hilfe eines bereits im Missionsplan vorhandenen horizontalen Transits unter Beachtung des maximalen Neigungswinkels überwunden werden kann. In diesem Fall wird das vorhandene Auf- oder Abtauchmanöver mit diesem Transit kombiniert, der dann auch den Ausgleich der vertikalen Positionsunterschiede zwischen Start- und Endpunkt vornimmt.

6.1.6.4 Beispiele und Ergebnisse

Die in der **Fehler! Verweisquelle konnte nicht gefunden werden.** aufgeführten Umplanungssituationen wurden beispielhaft mit dem Missionsplan aus Abbildung 6.30 untersucht. Dieser Plan befindet sich nicht in dem für die geografischen Planungsverfahren vorgesehenen Einsatzgebiet, da durch die in der Ostsee vorherrschende geringe Wassertiefe Einschränkungen bei den unterschiedlichen Umplanungen auftreten würden. Das genutzte Missionsgebiet besitzt demgegenüber eine ausreichende Wassertiefe, um auch mit Auftauch- und Abtauchmanövern operieren zu können. Für die Darstellung der Ergebnisse der geografischen Verfahren ist ein zweiter Missionsplan für das Gebiet der Eckernförder Bucht generiert worden, der am Ende dieses Abschnitts vorgestellt wird.

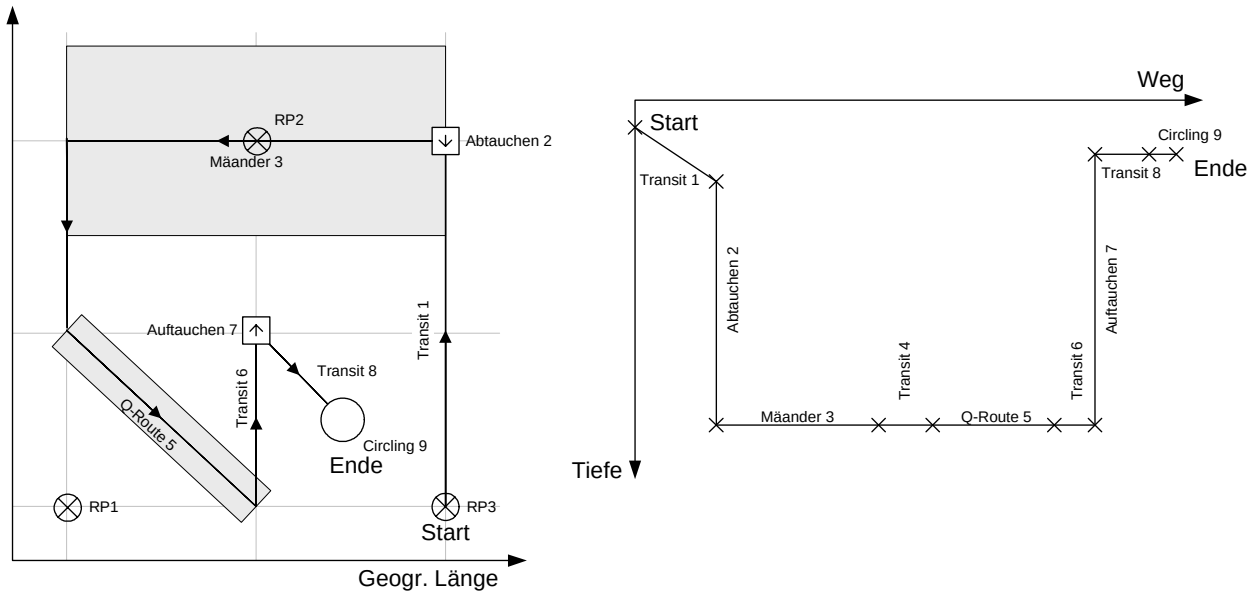


Abbildung 6.30: Missionsplan für die Tests des Missionsmanagements

Die verschiedenen Umplanungssituationen werden hier nicht einzeln dargestellt, sondern entsprechend der Gruppierung aus **Fehler! Verweisquelle konnte nicht gefunden werden.** zusammengefasst. Dadurch können ähnliche oder gleichartige Planmodifikationen gemeinsam erläutert werden.

6.1.6.4.1 Defekte der Sensorik

Ein Ausfall der Fahrzeugsensorik stellt unter Umständen einen kritischen Systemzustand dar. Insbesondere bei den primären Navigationssensoren ist deshalb ein Abbruch der aktuell gefahrenen Mission und, wenn der Missionsabbruch in einer vorgegebenen maximalen Entfernung um die geplanten Recovery-Positionen liegt, das Anfahren der nächsten dieser Positionen vorgesehen. Anhängig von der konkreten Situation sind unterschiedliche Manöver in den Missionsplan zu integrieren. In dem Beispiel aus Abbildung 6.31 sind nach dem Auftauchen ein Transit zu einer erreichbaren Recovery-Position und das abschließende Circling-Manöver als durchzuführende Folge von Planelementen erzeugt worden.

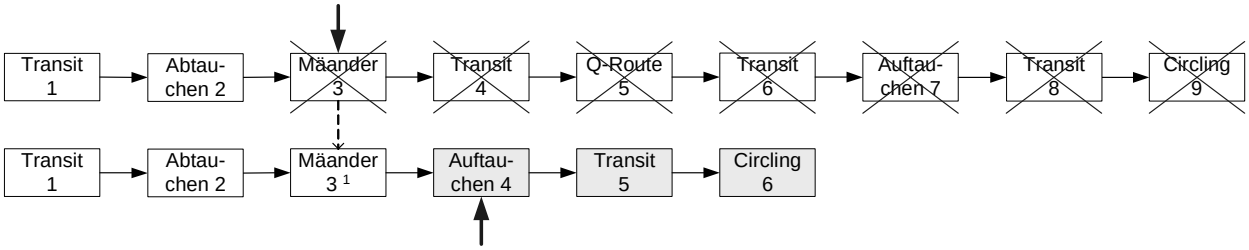


Abbildung 6.31: Beispiel für die Umplanung im Fall defekter Sensorik

Beim Ausfall der Nutzlast sind alle darauf angewiesenen Manöver aus dem Missionsplan zu entfernen. Durch die sich an die eigentliche Umplanung anschließende Planbereinigung werden dann die vorhandenen Transitmanöver zusammengefasst. Sind im überarbeiteten Plan noch zu realisierende Nicht-Transit-Manöver vorhanden, so kann die Mission mit deren Bearbeitung fortgesetzt werden. Beinhaltet der Plan hingegen lediglich Transitmanöver, wird durch das intelligente Missionsmanagement versucht, den geplanten Missionsendpunkt auf kürzestem Weg zu erreichen.

6.1.6.4.2 Ressourcenbezogene Umplanungen

Die *Optimierung des Missionsplanes* ist die bedeutendste ressourcenbezogene Umplanung. Sie wird gestartet, wenn ein Defizit beim Vergleich der vorhandenen Energiereserven mit den benötigten Ressourcen der noch zu realisierenden Manöver entdeckt wird. Die Missionsüberwachung fordert daraufhin eine Optimierung des Planes an, die von der Missionsumplanung umgehend realisiert wird. Aufgrund der in Abschnitt 6.1.4.3.2 dargestellten Komplexität des Optimierungsproblems und dem damit verbundenen hohen Zeitbedarf muss jederzeit ein Abbruch des Algorithmus möglich sein, um auf anderweitige, höherpriorisierte Umplanungsanforderungen reagieren zu können. Die abgebrochene Optimierung wird im Anschluss an die durchgeführte Planmodifikation neu gestartet, wenn das Ressourcendefizit noch vorhanden ist.

Aufgrund der Komplexität des Problems wurde neben dem eigentlichen Optimierungsverfahren eine Heuristik entworfen, die insbesondere bei großen Manöverzahlen eine erhebliche Beschleunigung des Optimierungsprozesses erzielt. Für einen Vergleich beider Methoden sind umfangreichere Pläne durch zwei-, fünf- und zehnfache Aneinanderreihung der Manöver 1 bis 8 des Testplanes aus Abbildung 6.30 erzeugt worden. Alle Pläne werden mit dem abschließenden Kontaktmanöver *Circling 9* beendet. Die Wiederholung der Manöversequenz stellt hohe Anforderungen an die Methoden der Planbereinigung, da das Entfernen von den in größeren Tiefen ausgeführten Manövern *Mäander 3* und *Q-Route 5* auch zur Kombination typischer Sequenzen von Abtauch-, Transit- und Auftauchmanövern zu einem einzelnen horizontalen Transit führen kann.

Den Maßstab für die Bewertung stellen der unter einem bestimmten Energiedefizit erreichte relative Wert der Mission $W_{rel}(\mathbf{x})$ bezogen auf den Wert des ursprünglichen Planes

$$W_{rel} = \frac{W\left(\mathrm{x}\right)}{W\left(\mathrm{x}^{*}\right)}$$

$$x_k^{*} = 1 \quad k = 1,\mathrm{K}\,,n$$

sowie die Laufzeit der Optimierung, gemessen über die Anzahl der Schritte bis zum Auffinden der Lösung, dar. Die Ergebnisse beider Optimierungsmethoden sind in Tabelle 6.6 gegenübergestellt. Die vermerkte Anzahl der Nicht-Transit-Manöver stellt dabei gleichzeitig den Umfang der zu optimierenden Aktivierungswerte und damit die Zahl der Optimierungsvariablen dar.

Tabelle 6.6: Ergebnisse und relativer Zeitbedarf der Missionsplanoptimierung

Missionsplan	Energie-defizit in %	Optimierungsverfahren Wert in %	Optimierungsverfahren Schritte	Heuristik Wert in %	Heuristik Schritte
Einfach	10,0	78,5	2	78,5	2
(9 Manöver, 2 Nicht-Transit)	25,0	78,5	2	78,5	2
	50,0	78,5	2	78,5	2
	75,0	0,0 [1]	2	0,0 [1]	2
Zweifach	10,0	89,2	2	89,2	4
(17 Manöver, 4 Nicht-Transit)	25,0	89,2	2	89,2	4
	50,0	78,5	4	78,5	4
	75,0	39,2	11	39,2	4
Fünffach	10,0	95,7	2	95,7	10
(41 Manöver, 10 Nicht-Transit)	25,0	91,4	7	91,4	10

	50,0	87,1	20	87,1	10
	75,0	78,5	62	78,5	10
Zehnfach	10,0	95,7	11	95,7	20
(81 Manöver, 20 Nicht-Transit)	25,0	91,4	140	91,4	20
	50,0	84,9	1637	84,9	20
	75,0	78,5	12024	78,5	20

[1] Der Missionsplan wird an der aktuellen Position abgebrochen, deshalb besitzt der verbleibende Anteil des Planes den Wert 0,0.

Im Vergleich der beiden Methoden wird deutlich, dass bei der verfügbaren Anzahl zu optimierender Manöver die Heuristik stets eine gleichwertige oder geringfügig schlechtere Lösung liefert als das Optimierungsverfahren. Diese Aussage lässt sich jedoch nicht verallgemeinern, da die Greedy-Heuristik im Gegensatz zu dem Optimierungsverfahren keine optimale Lösung garantiert. Durch die Anwendung der relativen Wertigkeiten wird jedoch stets ein gutes Ergebnis erreicht.

In den Fällen mit einem geringen Energiedefizit kann durch das Optimierungsverfahren die Lösung mit einer kleineren Anzahl von Schritten gefunden werden, da bedingt durch die Sortierung aller möglichen Manöverkonfigurationen die Lösung des Problems bereits nach wenigen Berechnungen vorliegt. Mit zunehmendem Defizit und damit auch mit zunehmendem Abstand der Lösung von der Startkonfiguration innerhalb der sortierten Liste erhöht sich der Berechnungsaufwand, bei großen Manöverzahlen sogar drastisch. Die Ursache dafür liegt in der bereits angesprochenen Komplexität des Problems.

Die Heuristik arbeitet im Gegensatz dazu mit einer der Anzahl an Optimierungsvariablen entsprechenden Schrittzahl. Bei steigender Manöverzahl oder größer werdendem Energiedefizit liegt die Laufzeit dieser Methode erheblich unter der des Optimierungsverfahrens. Die Qualität der Lösung, also der Wert des entstehenden Missionsplanes, ist dabei jedoch identisch oder geringfügig schlechter. Das Optimierungsverfahren erzielte in den Testbeispielen durch die Auswahl anderer Manöverkonfigurationen maximale Verbesserungen von ca. $2 \cdot 10^{-5}$ Prozent gegenüber der Heuristik.

Die Umschaltung zwischen den beiden Methoden erfolgt für das AUV DeepC abhängig von der zu optimierenden Manöveranzahl. Damit ist garantiert, dass eine Lösung in jeder Situation in möglichst kurzer Zeit berechnet werden kann. Gleichzeitig erlaubt dieses Umschaltkriterium eine einfache Anpassung an die verfügbare Rechenleistung der in einem autonomen mobilen System eingesetzten Rechentechnik, da sie maßgeblich die Berechnungsdauer beeinflusst.

Liegt eine zu *hohe Leistungsaufnahme des Gesamtsystems* vor, wird durch das Health-Monitoring eine Reduktion der Fahrgeschwindigkeit angefordert. Das Regelwerk des integrierten Expertensystems der Missionsüberwachung ermittelt aus der gewünschten Dauer der Reduktion die Anzahl der anzupassenden Manöver und erzeugt eine Reihe von atomaren Befehlen zum Modifizieren der Manövergeschwindigkeiten. Die ursprünglich geplanten Geschwindigkeitswerte werden als Fakten im Expertensystems gesichert, um sie nach Ablauf oder Rücknahme der Reduktion wieder anwenden zu können. Laufende Reduktionen können jederzeit hinsichtlich ihrer Dauer und zulässiger Fahrgeschwindigkeit parametriert werden. Damit steht ein flexibles Instrument zur Reaktion auf kurzfristige Energieengpässe zur Verfügung.

6.1.6.4.3 Anforderungen durch das Health-Monitoring

Kritische Situationen, die durch Ausfälle fahrzeugeigener Module oder die vorherrschenden Umgebungsbedingungen auftreten, werden durch das Diagnosesystem des Moduls Health-

Monitoring erkannt und bewertet. Da die Sicherheit des Fahrzeuges im Vordergrund aller Aktivitäten des Führungssystems steht, wird in der Regel ein Missionsabbruch kommandiert. Die Unterscheidung in einen unbedingten Abbruch und eine Auftauchempfehlung ermöglicht eine angemessene Reaktion in Abhängigkeit vom Grad der Gefährdung für das Fahrzeug.

6.1.6.4.4 Anforderungen durch weitere Module

Eine für die erfolgreiche Missionsdurchführung wesentliche Umplanungsaktion besteht in dem Einfügen von im Abschnitt 6.1.6.2 vorgestellten Event-Manövern. Diese Manöver werden bei der Missionsplanung erzeugt und bis auf die geografischen Positionen parametriert. Das intelligente Missionsmanagement passt die fehlenden Angaben an die zum Anforderungszeitpunkt existierenden Bedingungen an und fügt das Planelement an der aktuellen Position in den Missionsplan ein (Abbildung 6.32).

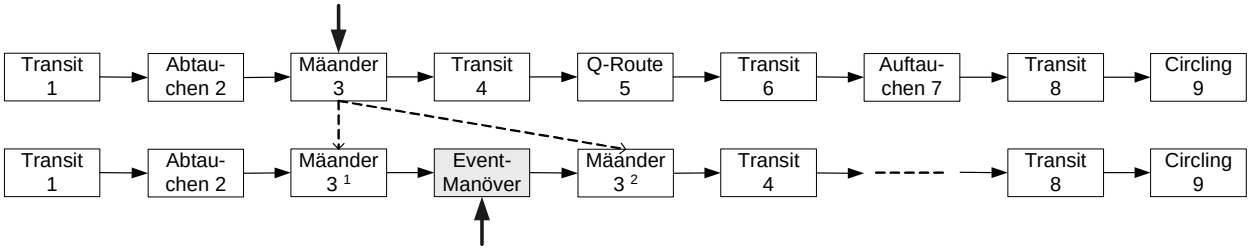


Abbildung 6.32: Einfügen eines Event-Manövers in den Missionsplan

Start- und Endposition können während der Planung mit relativen Verschiebungsvektoren belegt sein, die eine Verlegung des Event-Manövers bezogen auf die Einfügeposition erlauben (Abbildung 6.33 (a)). Beim Einfügen werden dann die absolute Position sowie die Ausrichtung des Manövers berechnet (Abbildung 6.33 (b)). Ist das Event-Manöver gegenüber der eigentlichen Einfügeposition verschoben, so hat zunächst ein Transit an die Startposition zu erfolgen (Abbildung 6.33 (c)). Analog dazu muss nach dem Ende des eingefügten Planelementes ein Transit zurück an die Einfügeposition durchgeführt werden. Ist das unterbrochene Manöver selbst ein Transit, so kann anstelle des Re-Transits zur Einfügeposition sofort die Endposition des unterbrochenen Manövers angefahren werden (Abbildung 6.33 (d)).

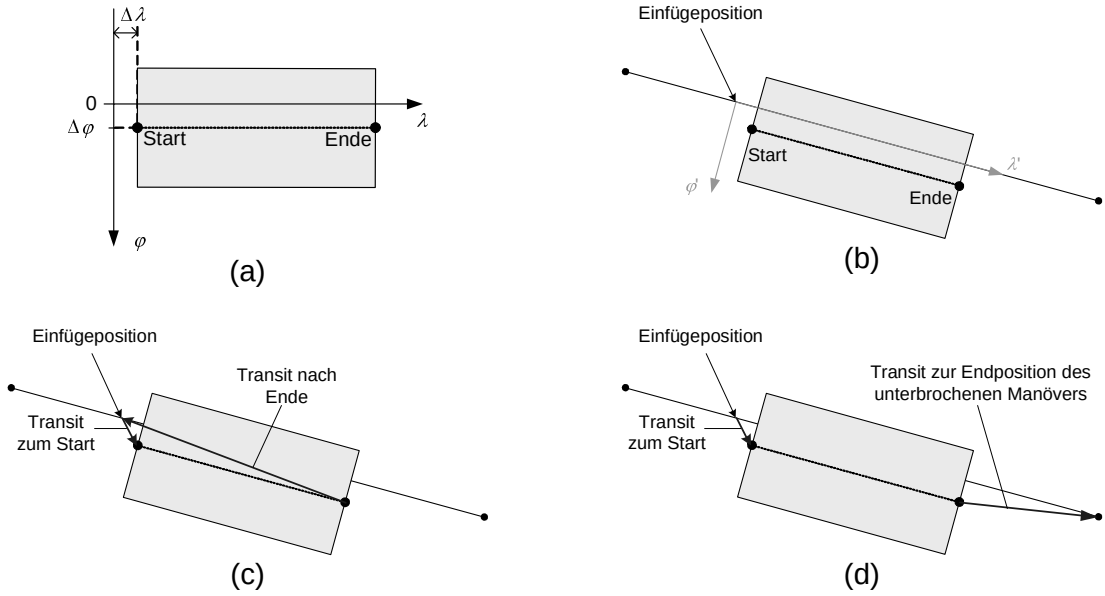


Abbildung 6.33: Berechnung der absoluten Koordinaten bei Event-Manövern ((a) und (b)), mögliche Anpassungen des Plans für Event-Manöver ((c) und (d))

6.1.6.4.5 Detektierte Abweichungen vom Missionsplan

Im Fall eines zu großen Navigationsfehlers wird durch die Missionsüberwachung das Einfügen eines GPS-Updates in den Missionsplan angefordert. Die auszuführenden Schritte bei der Missionsumplanung entsprechen den in der Abbildung 6.33 dargestellten Maßnahmen beim Einplanen eines Event-Manövers. Dabei wird das aktuelle Manöver unterbrochen, das GPS-Update eingefügt und anschließend der verbleibende Teil des noch nicht vollständig realisierten Manövers fortgesetzt.

Wird eine zu große vertikale Entfernung des AUV von der geplanten Fahrstrecke erkannt, muss ein Tiefenausgleich mit Hilfe eines vertikalen Transits durchgeführt werden. Die zugehörige Anforderung wird durch den Autopiloten kurz vor Beendigung eines Manövers erzeugt, wenn die geplante Endtiefe nicht erreicht werden kann. Die Missionsumplanung generiert daraufhin je nach Notwendigkeit ein Auf- oder Abtauchmanöver im Anschluss an das aktuelle Planelement.

6.1.6.4.6 Geografische Planprüfung und –modifikation

Die geografische Planprüfung wird im Anschluss an jede Umplanung durchgeführt, um eine Gefährdung des AUV durch die Modifikation des Planes auszuschließen. Als Beispiel zur Darstellung der Arbeitsweise der Planprüfung und einer sich im Fall auftretender Kollisionen anschließenden geografischen Planmodifikation wurde ein Missionsplan erstellt, der in der Eckernförder Bucht beginnt und in die offene See führt (Abbildung 6.34). Bei einer angenommenen Entfernung des zweiten Manövers *Q-Route 2* und der sich aus der anschließenden Planbereinigung ergebenden Zusammenfassung mit dem Manöver *Transit 1* entsteht eine nicht fahrbare Route für das AUV. Bei der geografischen Planprüfung werden dann zwei Schnittpunkte mit dem digitalen Höhenmodell festgestellt (Abbildung 6.35 (a)).

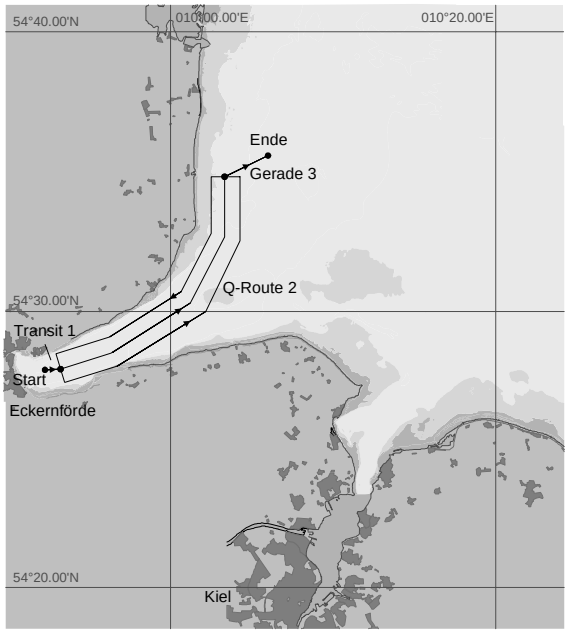
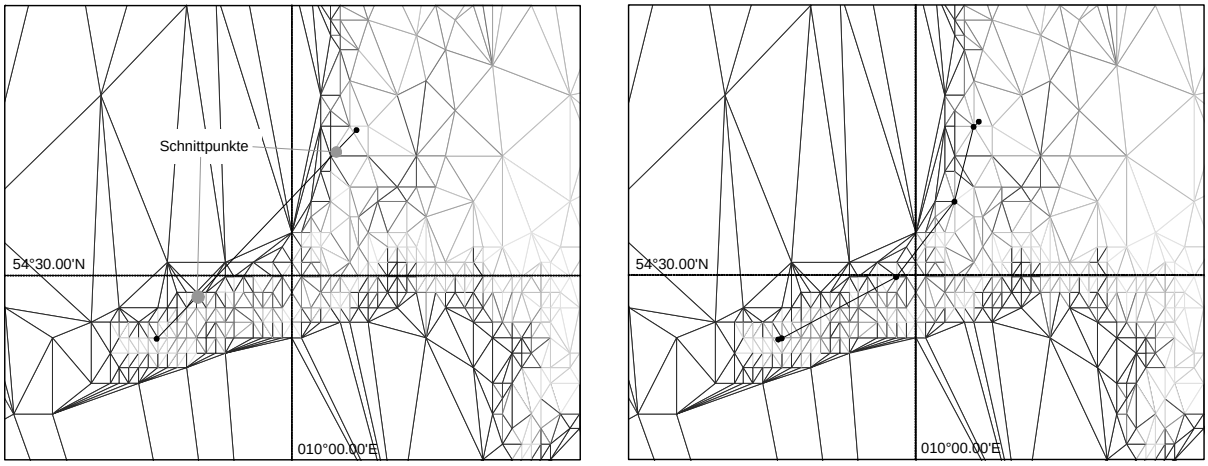


Abbildung 6.34: Missionsplan für den Test der geografischen Planungsverfahren

In einer derartigen Situation kann mit Hilfe der geografischen Planmodifikation eine fahrbare Route für diese Strecke berechnet werden. Das Ergebnis ist in Abbildung 6.35 (b) dargestellt. Zwischen dem eigentlichen Start- und Endpunkt des nicht fahrbaren Transits wird eine Folge von Stützstellen des digitalen Höhenmodells eingefügt, um eine manövrierbare Strecke zu generieren. Der ursprünglich vorhandene Transit wird dann durch eine Sequenz von Transit-Manövern zwischen den einzelnen Positionen entsprechend Abbildung 6.24

ersetzt und dieser gültige Missionsplan in die entsprechende Datenbank eingetragen. Alle den Plan nutzenden Module der Fahrzeugsoftware arbeiten ab diesem Zeitpunkt mit dem aktualisierten Missionsplan.



(a) (b)

Abbildung 6.35: Beispiel für die geografische Planprüfung (a) und -modifikation (b)

6.1.6.4.7 Weitere Umplanungssituationen

Neben den ausführlich dargestellten Umplanungen sind verschiedene weitere Bedingungen definiert, die eine Modifikation des Missionsplanes erfordern. Teilweise sind diese trivial oder können analog zu einer der geschilderten Vorgehensweisen ausgeführt werden. Deshalb soll auf eine Beschreibung derartiger Umplanungssituationen verzichtet werden.

6.1.7 Zusammenfassung

In diesem Kapitel wurde der Aufbau und die Anwendung eines Systems zum intelligenten Missionsmanagement für DeepC dargestellt. Durch die eingesetzte Softwarearchitektur ist eine nahezu transparente Integration in die Menge bereits bestehender Softwaremodule möglich.

Der Großteil der Ausführungen beschäftigte sich mit den theoretischen Grundlagen, die für die Umplanung einer Mission notwendig sind. Für DeepC wurde eine Menge von Umplanungssituationen definiert, die mit Hilfe des entworfenen Systems beherrscht werden müssen. Aufgrund der Vielfalt möglicher Umplanungsszenarien konnte mit den gezeigten Beispielen nur exemplarisch auf die wesentlichsten Methoden näher eingegangen werden. Die ausgewählten Situationen offenbaren jedoch die enorme Flexibilität des Konzeptes des intelligenten Missionsmanagements.

Die Optimierung des Missionsplanes stellt einen Schwerpunkt der entworfenen Verfahren dar. Durch die Implementierung zweier unterschiedlicher Methoden ist ein Einsatz direkt auf dem AUV ermöglicht worden. Der Vergleich von globaler Optimierung und Greedy-Heuristik zeigt die Leistungsfähigkeit des Näherungsverfahrens, wenn die Rahmenbedingungen sinnvoll gewählt werden.

Die geografischen Planungsverfahren garantieren schließlich die Fahrbarkeit der neu erstellten oder angepassten Manöver. Dazu ist neben der Prüfung eines Missionsplanes auch die Bestimmung eines navigierbaren Weges notwendig. Anhand eines im späteren ersten Einsatzgebiet des AUV DeepC gelegenen Missionsplanes konnte die prinzipielle Vorgehensweise der entworfenen Algorithmen und der Aufbau der Datenbasis dargestellt werden.

6.1.8 Literatur

[ACG+99] AUSIELLO, G. ; CRESCENZI, P. ; GAMBOSI, G. ; KANN, V. ; MARCHETTI-SPACCAMELA, A. ; PROTASI, M.: *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer Verlag, Berlin, Heidelberg, 1999

[BG93] BOMZE, I. M. ; GROSSMANN, W.: *Optimierung - Theorie und Algorithmen*. BI-Wissenschaftsverlag, Mannheim, Wien, Zürich, 1993

[BKI00] BEIERLE, C. ; KERN-ISBERNER, G.: *Methoden wissensbasierter Systeme*. Vieweg Verlagsgesellschaft mbH, Braunschweig, Wiesbaden, 2000

[BL99] BARROUIL, C. ; LEMAIRE, J.: Advanced Real-Time Mission Management for an AUV. In: *NATO Symposium on Advanced Mission Management and System Integration Technologies for Improved Tactical Operations*. Florence, Italy, sep 1999

[BW87] BÖHME, D. ; WERNSTEDT, J.: Entwurfskonzeote für Beratungssysteme zur Lösung kybernetischer Aufgaben. In: *messen - steuern - regeln (msr)* 30 (1987), 12, S. 535–539

[Dev98] DEVILLERS, O.: Improved Incremental Randomized Delaunay Triangulation. In: *Proceedings of the Fourteenth Annual Symposium on Computational Geometry*, 1998, S. 106–115

[Dij59] DIJKSTRA, E. W.: A Note on Two Problems in Connexion with Graphs. In: *Numerische Mathematik, Mathematisch Centrum, Amsterdam* 1 (1959), S. 269–271

[Dom95] DOMSCHKE, W.: *Einführung in Operations Research*. Springer-Verlag Berlin, Heidelberg, New York, 1995

[Doo95] DOORENBOS, R.: *Production Matching for Large Learning Systems*, Carnegie Mellon University, Pittsburgh, USA, Dissertation, 1995

[Dun95] DUNG, L. T.: *Entwurf und Realisierung von echtzeitfähigen Inferenztechniken für wissensbasierte Systeme*, Technische Universität Ilmenau, Dissertation, 1995

[For82] FORGY, C. L.: Rete: A Fast Algorithm for the Many Pattern / Many Object Pattern Match Problem. In: *Artificial Intelligence* 19 (1982), S. 17–37

[For94] FORGY, C. L.: RAL/C and RAL/C++: Rule-Based Extensions to C and C++. In: *OOPSLA Workshop*, 1994

[GJ79] GAREY, M. R. ; JOHNSON, D. S.: *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979

[GR98] GIARRATANO, J. ; RILEY, G.: *Expert Systems - Principles and Programming*. PWS Publishing Company, Boston, USA, 1998

[GRT93] GOERING, H. ; ROOS, H.-G. ; TOBISKA, L: *Finite-Element-Methode*. Akademie Verlag, Berlin, 1993

[Hau00] HAUN, M.: *Wissensbasierte Systeme*. expert verlag, Renningen, 2000

[Her89] HERTZBERG, J.: *Planen: Einführung in die Planerstellungsmethoden der künstlichen Intelligenz*. BI-Wissenschaftsverlag, Mannheim, Wien, Zürich, 1989

[HGM02] HAKE, G. ; GRÜNREICH, D. ; MENG, L.: *Kartographie*. Walter de Gruyter, Berlin, New York, 2002

[HJM93] HANSEN, P. ; JAUMARD, B. ; MATHON, V.: Constraint Nonlinear 0-1 Programming. In: *Journal on Computing, Operations Research Society of America* 5 (1993), Nr. 2, S. 97–119

[HRWL83] HAYES-ROTH, F. ; WATERMAN, D. A. ; LENAT, D. B.: *Building Expert Systems*. Addison-Wesley Publishing Company, Inc., Reading, USA, 1983

[Int00] INTERNATIONAL HYDROGRAPHIC ORGANIZATION: *IHO Transfer Standard for Digital Hydrographic Data - Publication S-57*. International Hydrographic Bureau, 2000

[Kni91] KNIERIEMEN, T.: *Autonome Mobile Roboter - Sensordateninterpretation und Weltmodellierung zur Navigation in unbekannter Umgebung*. BI-Wissenschaftsverlag, Mannheim, Wien, Zürich, 1991

[Lag96] LAGOUDAKIS, M. G.: The 0-1 Knapsack Problem, An Introductory Survey / Duke University, Department of Computer Science. 1996. – Forschungsbericht

[Leo96] LEONHARDT, B.: *Mission Plannung and Mission Control Software for the Phoenix AUV*, Naval Postgraduate School, Monterey, USA, Diplomarbeit, 1996

[Moh00] MOHAN, C. K.: *Frontiers of Expert Systems: Reasoning with Limited Knowledge*. Kluwer Academic Publishers, Norwell, USA, 2000

[MT90] MARTELLO, S. ; TOTH, P.: *Knapsack problems: algorithms and computer implementations*. John Wiley and Sons, New York, Chichester, Brisbane, Toronto, 1990

[MT97] MÖLLER, Tomas ; TRUMBORE, Ben: Fast, Minimum Storage Ray-Triangle Intersection. In: *Journal of Graphics Tools* 2 (1997), Nr. 1, S. 21–28. – ISSN 1086–7651

[Mur95] MURTY, K. G.: *Operations research: deterministic optimization models*. Prentice-Hall, Englewood Cliffs, 1995

[Nik97] NIKOLOPOULOS, C.: *Expert Systems: Introdcution to First and Second Generation and Hybrid Knowledge Based Systems*. Marcel Dekker, Inc., New York, 1997

[Pri00] PRITCHARD, D. E.: *Dynamic Route Replanning and Retasking of Unmanned Aerial Reconnaissance Vehicles*, Air Force Institute of Technology, Wright-Patterson AFB Ohio, USA, Diplomarbeit, 2000

[PS90] PREPARATA, F. P. ; SHAMOS, M. I.: *Computational geometry: An Introduction*. Springer Verlag, New York, 1990

[Ree95] REEVES, C. R. (Hrsg.): *Modern heuristic techniques for combinatorial problems*. McGraw-Hill, London, 1995

[SMK95] SILVA, C. T. ; MITCHELL, J. S. B. ; KAUFMAN, A. E.: Automatic Generation of Triangular Irregular Networks using Greedy Cuts. In: *Proceedings of the 6th IEEE Visualization 1995 Conference (VIS ’95)*, 1995

[STK01] SEIFERT, T. ; TAUBER, F. ; KAYSER, B.: A high resolution spherical grid topography of the Baltic Sea - revised edition. In: *Proceedings of the Baltic Sea Science Congress*, 2001

6.2 Führung des Fahrzeuges in Sondersituationen (FIS)

6.2.1 Einbindung des Moduls FIS in die Führungshierarchie des Fahrzeuges

Die Führungshierarchie des Fahrzeuges ist in eine Mehrebenenstruktur [BHMM97] aufgebaut (siehe Abbildung 6.36). Im Normalfall arbeitet das Fahrzeug einen Missionsplan ab. Dieser Plan enthält Positionswerte und geometrische Daten zur Beschreibung von Basismanövern, welche das Fahrzeug in einer vorgegebenen Reihenfolge abzufahren hat. Dabei werden definierte Vorgaben zur Führung des Fahrzeuges in Form von Sollkurs, -tiefe, -bahn, -lage und -geschwindigkeit an den Autopiloten übergeben. Dies ist vergleichbar mit der vollautomatischen Führung eines Passagierflugzeuges durch einen Flugmanagementrechner.

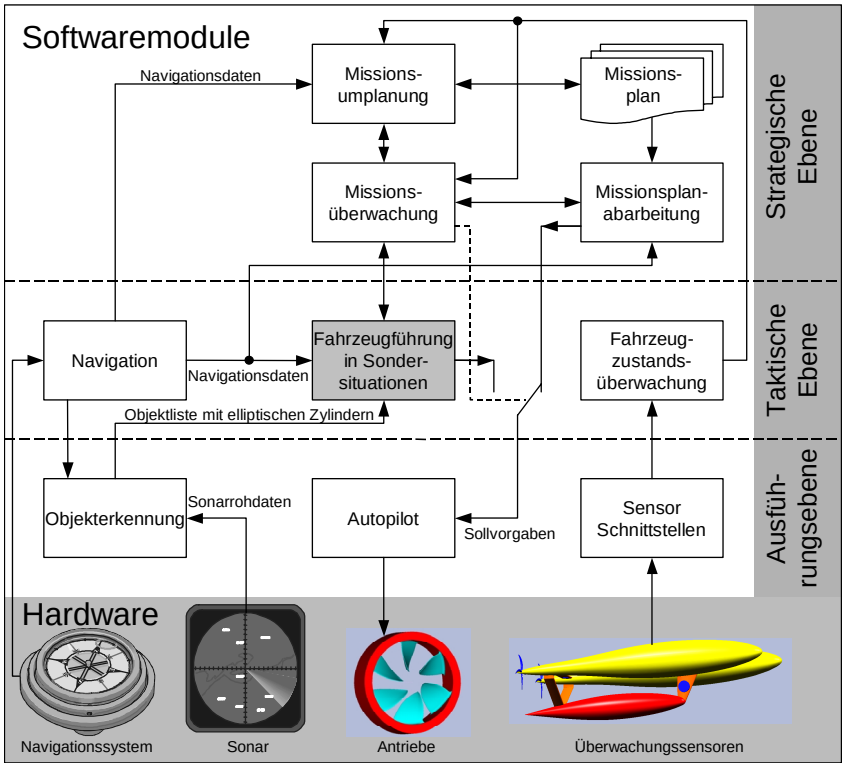


Abbildung 6.36: Steuerungsstruktur des Fahrzeuges

Das Softwaremodul *Objekterkennung* detektiert aus den Rohdaten des Sonarsystems Objekte in Form von elliptischen Zylindern. Diese Form lässt sich leicht aus den Rohdaten des Sonars generieren und charakterisiert durch wenige geometrische Abmaße (Hauptachse, Nebenachse und Rotationswinkel der Ellipse, Höhe des Zylinders) eine Vielzahl von praktischen Hindernisstrukturen. Tangieren solche Objekte die Sollbahn des Fahrzeuges, wird die automatische Führung durch andere Algorithmen abgelöst. In Abhängigkeit der Missionsaufgabe kann die Aufgabe dieser Algorithmen die Führung des Fahrzeuges zum Ausweichen oder zum Identifizieren der Objekte sein. Diese Algorithmen sind Hauptkomponenten des Softwaremoduls „Fahrzeugführung in Sondersituationen“ (FIS) (in Abbildung 6.36 grau hinterlegt) und werden in den nachfolgenden Abschnitten detailliert vorgestellt.

Bei der Sondersituation *Ausweichen* wird das Fahrzeug vorbei an den Hindernissen zu einem vorgegebenen Zielpunkt geführt, welcher auf dem aktuellen Missionsplan liegt. Eine weitere Sondersituation besteht in der *Identifikation* von unbekannten Objekten während einer Mission. Eine Identifikation kann eine visuelle oder kartographische Erfassung des Objektes und/oder eine Bestimmung seiner physikalischen/chemischen Eigenschaften

beinhalten. Die Steuerung des Fahrzeuges wird hierbei durch Algorithmen zur Positionierung und Führung des Fahrzeuges übernommen.

Um bei den eingangs beschriebenen Vergleich zu bleiben, wären diese beiden Sondersituationen (Hindernisvermeidung, Identifikation) in Analogie mit der manuellen Führung eines Flugzeuges durch den Piloten zu sehen.

6.2.2 Aufbau und Funktionsweise des Moduls FIS

Das Modul FIS besteht aus den nachfolgenden Hauptkomponenten (Abbildung 6.37):

- Kollisionsüberwachung
- Zielpunktgenerierung
- Hindernisvermeidungssystem
- Identifikationssystem
- Regelungsalgorithmen.

Die Objektinformationen werden vom Softwaremodul *Objekterkennung* in Form von elliptischen Zylindern an das Softwaremodul FIS übergeben. Die *Kollisionsüberwachung* überprüft während der Mission eine Kollisionsmöglichkeit zwischen den detektierten Objekten und den aktuellen Manövern des Missionsplanes. Ist eine solche Möglichkeit gegeben, wird dies an das Modul Missionsüberwachung gemeldet. Dieses Modul aktiviert in Abhängigkeit des Missionsplanes das *Hindernisvermeidungssystem* oder das *Identifikationssystem*. Das aktivierte System übernimmt dann die Führung des Fahrzeuges und übergibt die Steuerinformationen des aktiven Systems an das Submodul *Regelungsalgorithmen*. Solche Steuerinformationen können Vorgaben für Solltrajektorien, anzufahrende Positionen oder der Abstand zu einem Objekt sein. Im Submodul *Regelungsalgorithmen* werden diese Steuerinformationen so aufbereitet, dass sie an den Autopiloten in Form von Sollwerten für Kurs, Tiefe und Geschwindigkeit übergeben werden. Die *Zielpunktgenerierung* berechnet während einer solchen Sondersituation einen Rendezvouspunkt mit der Sollroute des Missionsplanes unter Verwendung des aktuellen Missionsplanes und der detektierten Objekte.

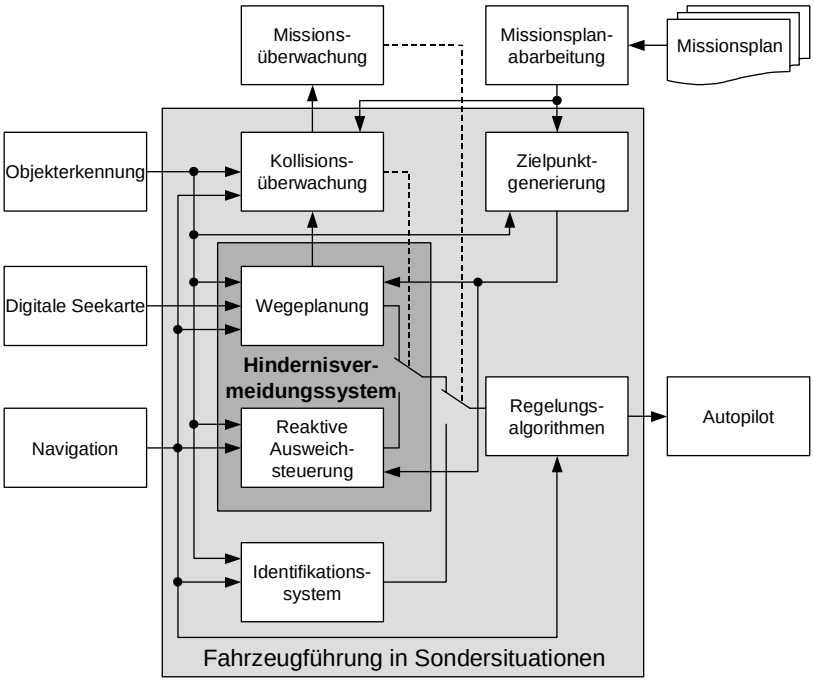


Abbildung 6.37: Struktur des Softwaremoduls FIS

Die einzelnen Komponenten des Moduls FIS werden in den nachfolgenden Abschnitten detailliert vorgestellt.

6.2.3 Kollisionsüberwachung

6.2.3.1 Einführende Erläuterungen

Für das bessere Verständnis des Zusammenspiels des Moduls FIS mit den anderen Modulen sollen eingangs dieses Abschnittes einige einführende Erläuterungen dargelegt werden.

Aufbau eines Missionsplanes

Ein Missionsplan setzt sich aus einer Folge von Manövern zusammen. Diese Manöver sind in Komplexmanöver und Übergangsmanöver unterteilt. Komplexmanöver erfüllen eine spezielle Aufgabe (Abtauchen, Mäander, GPS-Update) innerhalb des Missionsplanes. Die Übergangsmanöver dienen als Verbindung zwischen End- und Startposition zweier Komplexmanöver. Abbildung 6.38 zeigt einen Missionsplan mit einzelnen Manövern.

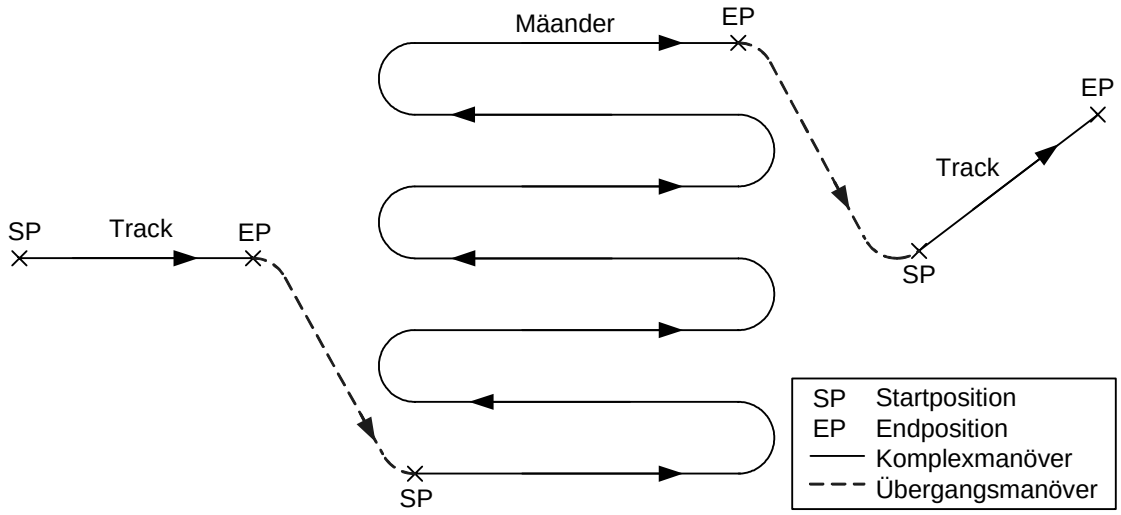


Abbildung 6.38: Missionsplan

Diese einzelnen Manöver setzen sich aus Basismanövern zusammen.

Beschreibung der Basismanöver

Basismanöver stellen einfache geometrische Linienverläufe dar, welche durch eine Start- und Endposition, geometrische Daten zum Beschreibung ihrer Form und Sollgeschwindigkeitsvorgaben charakterisiert sind. Nach [Fos02] umfasst die Lösung einer Manöveraufgabe zwei Teile:

Geometrische Aufgabe: Führung des Fahrzeuges auf der vorgegebenen Solltrajektorie.

Dynamische Aufgabe: Steuerung des Fahrzeuges nach den Sollgeschwindigkeitsvorgaben.

Es werden die vier Basismanöver:

- Strecke
- Kreisbogen
- Kreis
- Kreuzspirale

definiert (siehe Abbildung 6.39).

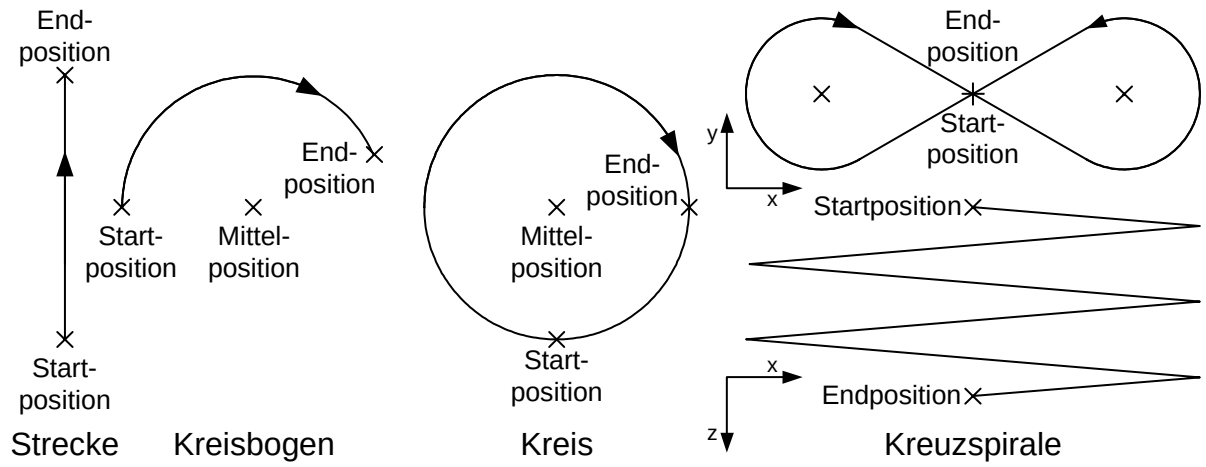


Abbildung 6.39: Basismanöver

Diese Basismanöver bilden somit die Grundbausteine eines Missionsplanes. Aus den Komplex- bzw. Übergangsmanövern wird vom Modul Missionsplanabarbeitung das aktuelle Basismanöver herausgelöst und an den Autopiloten übergeben (siehe Abbildung 6.36). Dort werden die beiden o.g. Manöveraufgaben unter Vorgabe der geometrischen und dynamischen Parameter durch entsprechende Regelungen gelöst. Abbildung 6.40 zeigt den Aufbau eines Missionsplanes mit Basismanövern aus dem in Abbildung 6.38 dargestellten Missionsplan.

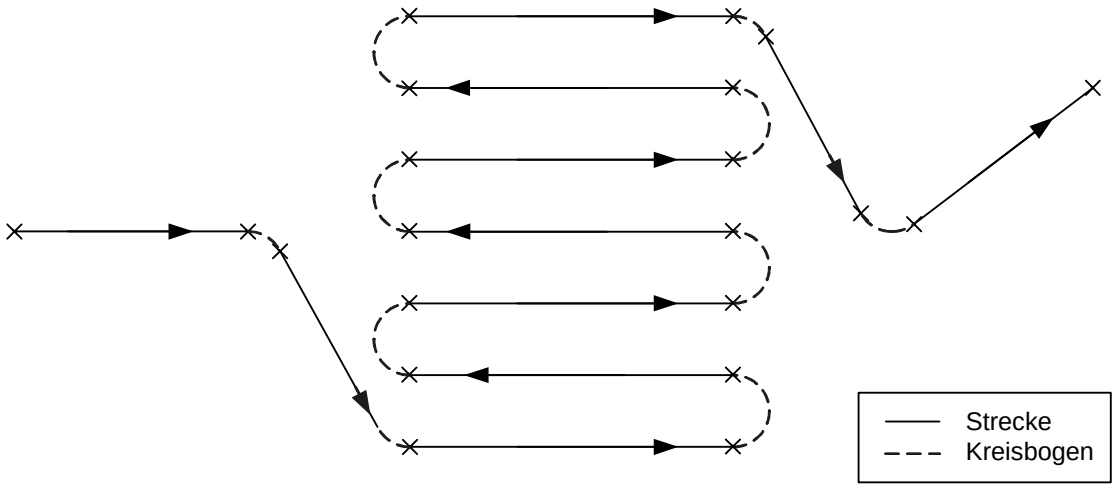


Abbildung 6.40: Basismanöverplan

Das Submodul Kollisionsüberwachung fordert von der Missionsplanabarbeitung einer Liste des aktuellen Basismanövers und der n-nächsten Basismanövern an. Die einzelnen Manöver dieser Liste werden dann auf Kollision mit allen detektierten Objekte überprüft.

Definition von Umgebungsbereichen um die Objekte

Durch die geometrischen Abmaße des Fahrzeuges, seine mögliche Ablage zur Sollbahn sowie einer Varianz der Objektpositionen und -geometrien müssen die detektierten Objekte um einen Bereich erweitert werden. Dies erfolgt durch die Vergrößerung der Haupt- und Nebenachse der elliptischen Grundfläche sowie der Erhöhung der zylindrischen Objektform um einen vorgegebenen Wert. In Abhängigkeit der Verwendung der Objekte in den einzelnen Submodulen wurden zwei Werte (Abstände zum Objekt) definiert. Für das Submodul *Hindernisvermeidungssystem* wird ein *Sicherheitsabstand* vorgegeben. Die Notwendigkeit zur Einführung eines solchen Sicherheitsbereich um das Objekt hat neben

den oben genannten noch einen weiteren Grund. Die beim Hindernisvermeidungssystem generierten Bahnen sind wegoptimal, was dazu führt, dass sie die Objekte tangieren können. Der Sicherheitsbereich verhindert so ein zu nahes Vorbeifahren am Objekt.

Ein definierter *Manöverabstand* kommt bei den Submodulen *Kollisionsüberwachung* und *Zielpunktgenerierung* zum Einsatz. Die um diesen Wert vergrößerten Objektformen werden zur Berechnung von Kollisionssituationen und zur Bestimmung des anzufahrenden Wegpunktes verwendet. Abbildung 6.41 zeigt die definierten Bereiche um ein Objekt. Bei den Darlegungen in den weiteren Abschnitten wird davon ausgegangen, dass die Objektgeometrie um den jeweiligen Bereich vergrößert wurde.

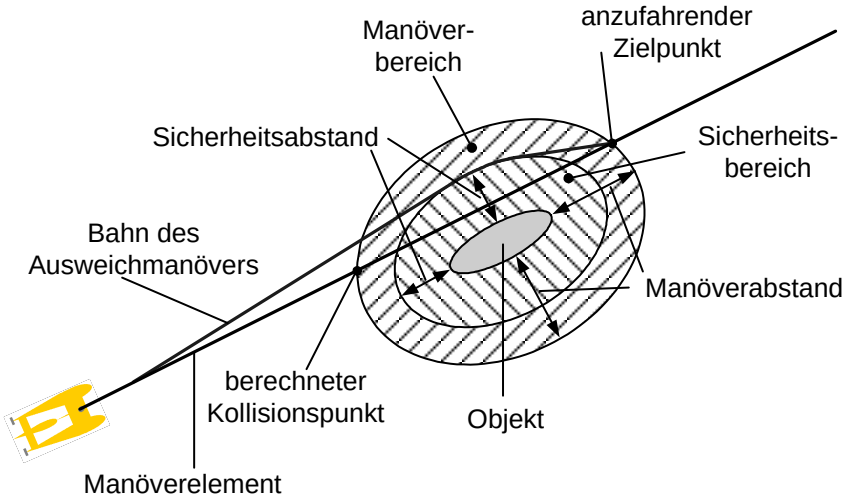


Abbildung 6.41: Definition der Sicherheitsbereiche

6.2.3.2 Ermittlung einer Kollisionssituation

Die Aufgabe des Moduls Kollisionsüberwachung besteht in der Bestimmung der nächsten Kollisionssituation des Fahrzeuges mit einem Objekt auf der Basis der Fahrtroute und der detektierten Objekte. Die in diesem Zusammenhang interessierenden Parameter sind die Kollisionsposition und die Zeitdauer bis zur Kollision.

Bei der Kollisionsüberprüfung werden die $m_{Manöver}$ Basismanöver mit den $n_{Objekte}$ detektierten Objekten auf Überschneidungen abtestet. Die Anzahl durchzuführenden Test num_{Test} beträgt so

$$num_{Test} = m_{Manöver} \cdot n_{Objekte} . \tag{6.5}$$

Die Überprüfung auf Kollision erfolgt in einem zweischleifigen Prozess. Jedes einzelne Manöver wird nach der Reihenfolge seiner Abarbeitung im Missionsplan gegen die gesamte Liste der Objekte abgeprüft. Gibt es eine Kollision zwischen dem Manöver und einem Objekt, wird die Zeit bis zur Kollision t_{Koll} und die Kollisionsposition X_{Koll} abgespeichert. Tritt eine weitere Kollisionssituation dieses Manövers mit einem Objekt aus der Objektliste auf, werden die Kollisionsparameter übernommen, wenn die Zeit zur Kollision kürzer als die bis jetzt abgespeicherte Zeit ist. Wenn bei der Überprüfung eines Manöver mit der gesamten Objektliste eine Kollision auftrat, bracht die Überprüfung der restlichen Manöverelemente nicht mehr durchgeführt zu werden. Die gefundenen Kollisionsparameter werden ausgegeben.

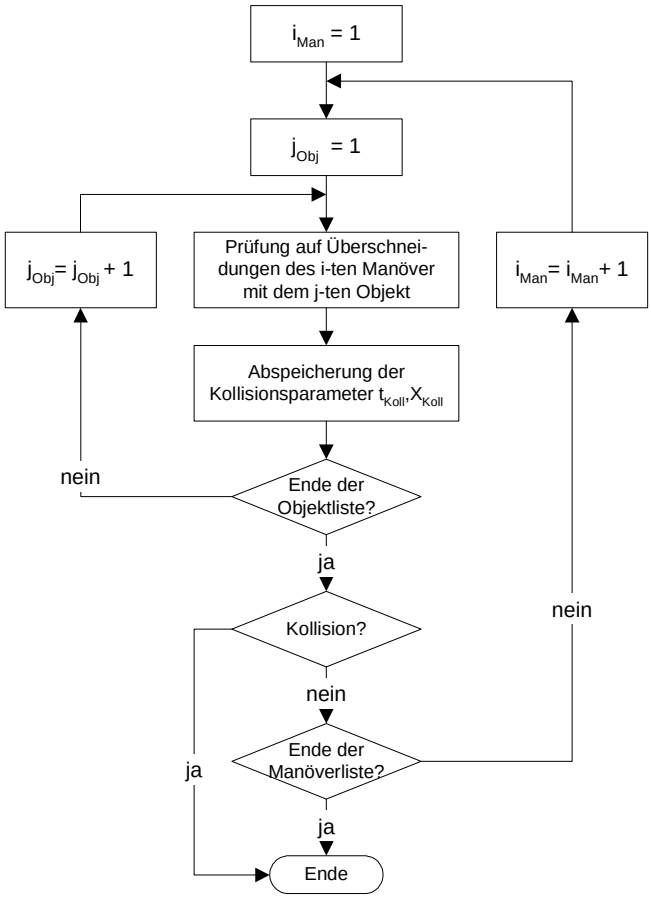


Abbildung 6.42: Bestimmung der ersten Kollisionssituation

Die Überprüfung auf Überschneidung der Objekte mit den Manövern erfolgt durch Schnittberechnungen der geometrischen Formen von Objekt und Manöver. Diese sind für den 3-dimensionalen Fall:

- Schnitt Elliptischer Zylinder – Strecke
- Schnitt Elliptischer Zylinder – Kreisbogen
- Schnitt Elliptischer Zylinder – Kreuzspirale.

Durch diese mathematisch aufwendigen Schnittberechnungen [Ebe01], [SE03], [Wei03] kann nach Gleichung (6.5) die Gesamtrechenzeit zur Ermittlung einer Kollisionssituation bei einer großen Manöver- und/oder Objektanzahl sehr hoch sein.

Aus diesem Grund und durch die geringe Wahrscheinlichkeit von Überschneidungen zwischen den Manövern und Objekten innerhalb des Operationsgebietes wurde ein hierarchisches Konzept zur Bestimmung einer Kollisionssituation entworfen. Die Idee besteht in der Nachbildung der geometrischen Formen der Manöver und Objekte durch Begrenzungsboxen, auf deren Basis einfache Schnittüberprüfungen durchgeführt werden können. Erst wenn diese Überprüfungen erfolgreich waren, brauchen die exakten Schnittberechnungen durchgeführt zu werden. Eine Überlappung der Begrenzungsboxen bildet somit eine notwendige Bedingung für einen möglichen Schnitt zwischen Manöver und Objekt.

In einer ersten Stufe wird eine Begrenzungsbox in Form eines achsenparallelen Rechteckes bzw. für den 3D-Fall eines achsenparallelen Quaders für das Manöver und das Objekt definiert [Mir98], [SE03]. In Abbildung 6.43 sind diese Begrenzungsboxen für die einzelnen

Manöver und Objekte eines möglichen Missionsszenarios dargestellt. Die Anzahl der hierbei durchzuführenden Tests beträgt bei den 10 Basismanövern und den 4 Objekten 40.

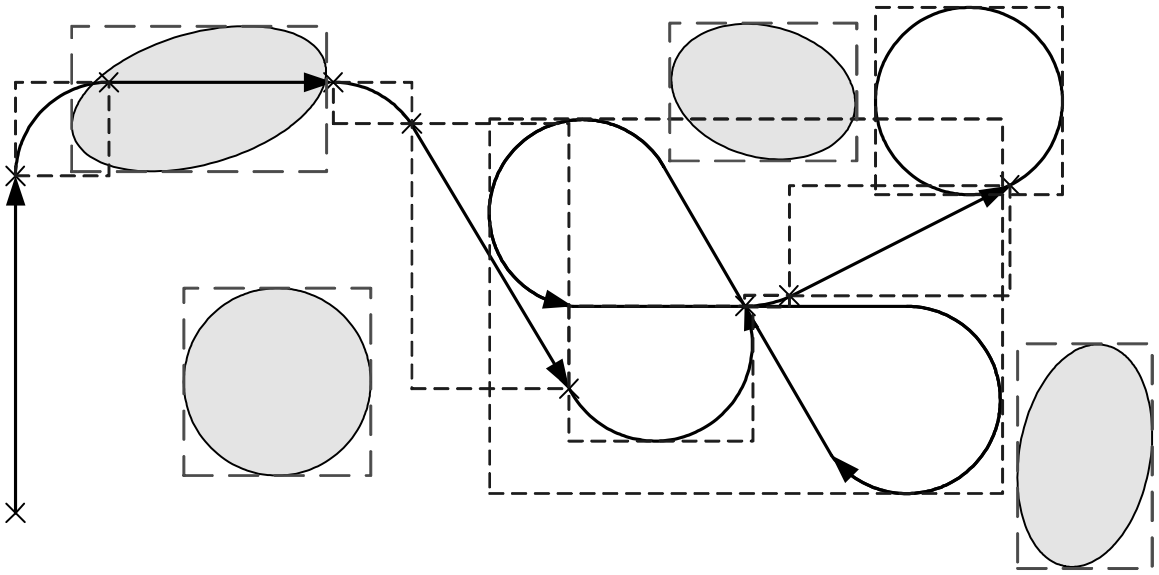


Abbildung 6.43: Beschreibung der Manöver und Objekte durch achsenparallele Rechtecke

Die achsenparallele Begrenzungsbox wird durch die Punkte $\mathbf{P_{min}}=[x_{min}, y_{min}, z_{min}]$ und $\mathbf{P_{max}}=[x_{max}, y_{max}, z_{max}]$ beschrieben (siehe Abbildung 6.44).

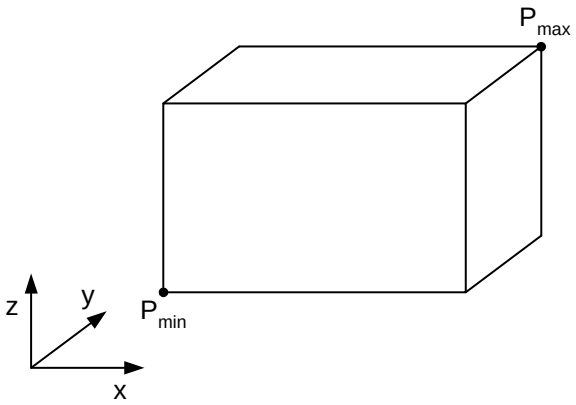


Abbildung 6.44: Definition der achsenparallelen Begrenzungsbox

Eine Überprüfung erfolgt für die einzelnen Koordinaten durch den gegenseitigen Größenvergleich des maximalen Wertes der einen mit den minimalen Wert der zweiten Box.

Der Pseudocode dafür lautet:

```
bool ExistiertSchnittBoxBox(Box Box1,Box Box2)
{
  if (Box1.max.x) < (Box2.min.x) return false;
  if (Box2.max.x) < (Box1.min.x) return false;
  if (Box1.max.y) < (Box2.min.y) return false;
  if (Box2.max.y) < (Box1.min.y) return false;
  if (Box1.max.z) < (Box2.min.z) return false;
  if (Box2.max.z) < (Box1.min.z) return false;
  return true;
}
```

Da bei diesem ersten Test nur Vergleichoperationen durchgeführt werden, ist der benötigte Rechenaufwand sehr minimal. Durch diesen Test kann die mögliche Anzahl von Überschneidungen zwischen Manövern und Objekten sehr eingeschränkt werden. Für das in Abbildung 6.43 dargestellte Szenario wurden zwei mögliche Überscheidungssituationen ermittelt. Nur für diese beiden Überscheidungssituationen wird ein weiterer Test durchgeführt.

Bei diesem zweiten Test werden die Begrenzungsboxen im Raum so ausgerichtet, dass sie die kleinste Fläche bzw. das kleinste Volumen besitzen und dabei das Manöver bzw. das Objekt vollständig einschließen. Es werden die Tests

- Schnitt Box – Box
- Schnitt Strecke – Box

durchgeführt. In Abbildung 6.45 sind die Begrenzungsboxen der einzelnen noch zu untersuchenden Manöver und Objekte dargestellt. Für diesen Test wurde eine Überscheidungssituation ermittelt.

So konnte, durch die im Vorfeld durchgeführten vereinfachten Schnittberechnungen mit den Begrenzungsboxen, der Berechnungsaufwand von vierzig auf eine exakte Schnittberechnung gesenkt werden.

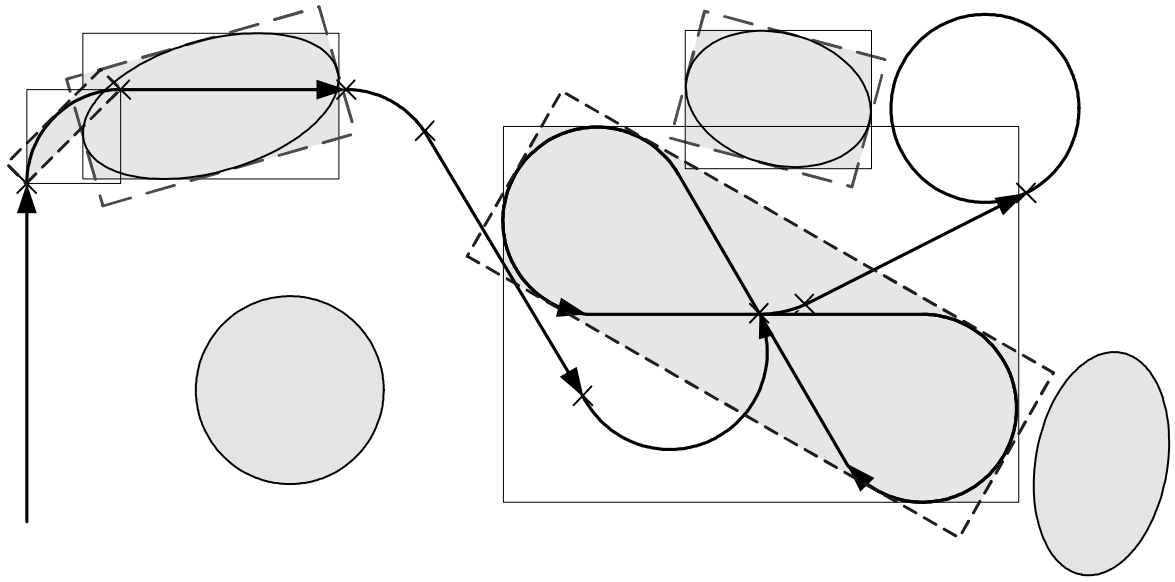


Abbildung 6.45: Beschreibung der Manöver und Objekte durch ausgerichtete Begrenzungsboxen

6.2.4 Zielpunktgenerierung

Das Submodul *Zielpunktgenerierung* berechnet während einer Sondersituation (Ausweichen oder Identifikation) unter Verwendung des aktuellen Missionsplanes und der detektierten Objekte, einen Rendezvouspunkt mit der Sollroute des Missionsplanes. Dieser Punkt wird dabei so geplant, dass möglicht wenig Verluste an durchzuführenden Missionsaufgaben durch nicht befahrene Manöverelemente entstehen.

Im Ausweichmodus entspricht dieser Punkt den anzufahrenden Zielpunkt. (siehe Abschnitt 6.2.5) Bei aktiven Identifikationsmodus wird nach Durchführung der Identifikationsaufgabe intern im Modul FIS in den Zustand Ausweichmodus gewechselt. Das Fahrzeug wird so wieder auf den aktuellen Missionsplan zurückgeführt. Kreuzt bei dieser Rückführung ein

noch nicht identifiziertes Objekt die Bahn des Fahrzeuges wird in den Zustand Identifikationsmodus geschaltet und es findet die Identifikation des Objektes statt.

Die Idee zur Bestimmung des Zielpunktes basiert darauf, dass die aktuelle Liste von m Manövern vom Endpunkt aus „Durchfahren“ wird. Die erste bestimmte „Kollisionssituation“ entspricht den anzufahrenden Zielpunkt. Die hierfür notwendigen Berechnungen verwenden die gleichen Algorithmen wie das Submodul *Kollisionsüberwachung* (siehe Abschnitt 6.2.3.2.). Durch die Anzahl m an zu überprüfenden Manövern kann direkt Einfluss auf einen hohen Erfüllungsgrad an Missionsaufgaben genommen werden. Eine hohe Manöveranzahl m bewirkt eine langen Missionsabschnitt. Wird gleich zu Beginn (am Ende des Missionsabschnittes) der zu überprüfenden Manöverliste eine Kollision ermittelt, wird das Fahrzeug zu diesem Punkt geführt. Die vorhanden Manöverabschnitte bis zu diesem Zielpunkt werden nicht mehr abgefahren. Ihre Missionsaufgaben wurden so nicht erfüllt. Diese Überlegungen sind von besonderen Interesse beim Durchfahren eines Mäanders. Hier bewirkt eine hohe Anzahl an zu überprüfenden Manövern gegebenenfalls das Nichtbefahren von Suchgebieten (wenn ein Hindernis mehrere Streckenabschnitte des Mäanders schneitet). Abbildung 6.46 zeigt den Algorithmus zur Bestimmung des Zielpunktes.

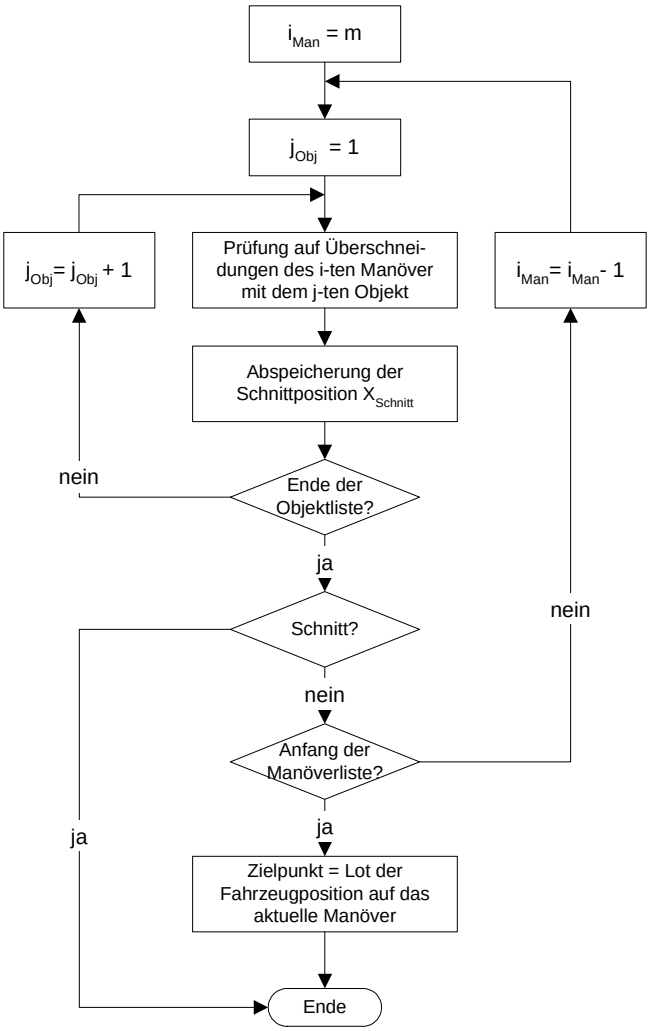


Abbildung 6.46: Bestimmung des Zielpunktes

6.2.5 Hindernisvermeidungssystem

Für das Hindernisvermeidungssystem des Fahrzeuges wurde eine Zweiebenen - Struktur favorisiert (im Abbildung 6.37 dunkelgrau hinterlegt). Die obere Ebene verwendet zusätzlich zu den aktuellen Informationen des Sonars die gesammelten Hindernisdaten der bisherigen Mission sowie die Daten einer digitalen Seekarte. Mit diesen Informationen findet eine *Wegeplanung* auf der Basis geometrischer Graphen statt. Die Verfahren hierzu ermöglichen es, einen optimalen Weg nach definierten Vorgaben in einer kalkulierbaren Zeit zu ermitteln, welchem das Fahrzeug folgt. Einen Überblick über die bei der *Wegeplanung* eingesetzten Verfahren gibt Abschnitt 6.2.5.1.

Treten beim Abfahren des generierten Weges unvorhersehbare, plötzlich auftauchende Hindernisse auf, übergibt die *Kollisionsüberwachung* die Steuerung an die untere Ebene, die *Reaktive Ausweichsteuerung*. Die *Reaktive Ausweichsteuerung* reagiert auf die im Nahbereich des Sonars aufgefassten Hindernisse durch entsprechende reaktive Steuerkommandos, wodurch ein wegoptimales Umfahren nicht immer möglich ist. Eine solche Forderung steht hier aber nicht an oberster Priorität. Aufgabe ist ein sicheres und schnelles Ausweichen zur Vermeidung einer Kollisionssituation. In der Aktivierungszeit der Ebene *Reaktiven Ausweichsteuerung* kann in der Ebene *Wegeplanung* der Routenplan unter Verwendung der neuen Objektinformationen modifiziert bzw. neu erstellt werden. Das Ausweichen mit der *Reaktiven Ausweichsteuerung* erfolgt nur in der x-y Ebene. Dies ist durch den begrenzten Nicklagewinkel des Fahrzeuges infolge seiner konstruktiven Gegebenheiten und die gute Manövrierfähigkeit in der horizontalen Ebene begründet. Eine detaillierte Beschreibung der in der *Reaktiven Ausweichsteuerung* eingesetzten Algorithmen vermittelt Abschnitt 6.2.5.2.

Abbildung 6.47 stellt generierte Fahrtrouten bei Verwendung der *Wegeplanung* und der *Reaktiven Ausweichsteuerung* am Beispiel eines möglichen Hindernisparcours bei gegebenen Start- und Zielpunkt gegenüber. Hier ist deutlich das reaktive („lokale“) Verhalten der *Reaktiven Ausweichsteuerung* und das „vorausschauende“ („globale“) Verhalten der *Wegeplanung* an Punkt A zu erkennen. Die *Wegeplanung* nutzt alle Objektinformationen zur Generierung einer Route, wogegen die *Reaktive Ausweichsteuerung* nur die im Nahbereich vorhandenen Objekte zur Generierung der Sollkommandos verwendet.

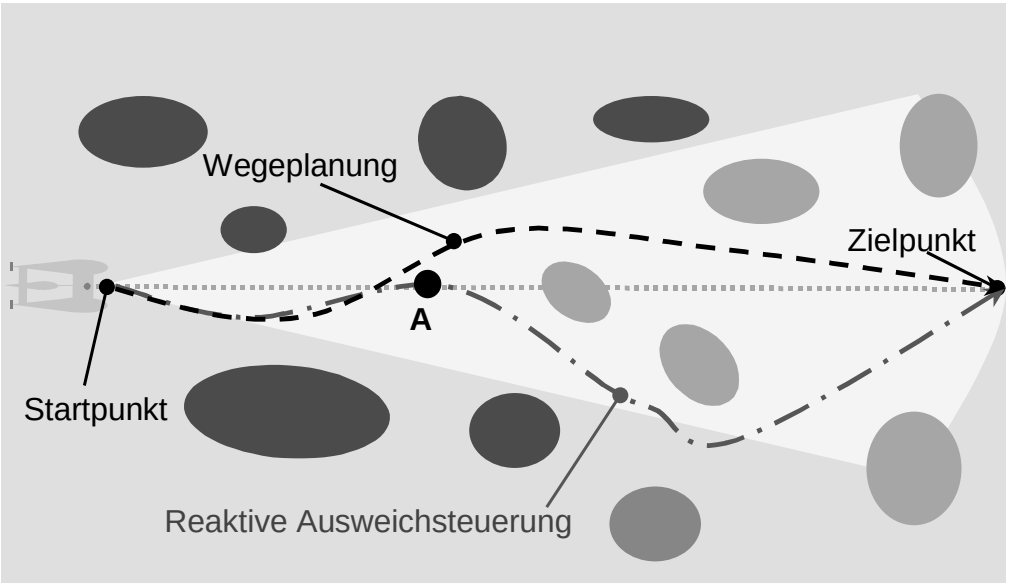


Abbildung 6.47: Fahrtrouten bei Verwendung der einzelnen Ebenen

6.2.5.1 Wegeplanung

Bei der Wegeplanung werden die gesamten Informationen über das aktuelle Operationsgebiet zur Generierung eines Routenplanes verwendet. Das sind neben den aktuellen Informationen des Sonars, die „gesammelten“ Hindernisdaten der bisherigen Mission sowie die Daten einer digitalen Seekarte. Ein solcher Routenplan wird auf Basis von Graphen ermittelt. Dabei werden Punkte (Knoten) im Operationsgebiet definiert, welche durch das Fahrzeug befahrbar sind. Die befahrbaren Verbindungen zwischen diesen Punkten werden als Kanten in den Graphen eingetragen. Jede Kante besitzt eine Bewertung (Kosten, Gewicht), welche die Länge der Verbindung, die entstehenden Kosten beim Abfahren der Verbindung oder die dafür benötigte Zeit sein kann (siehe Abbildung 6.48).

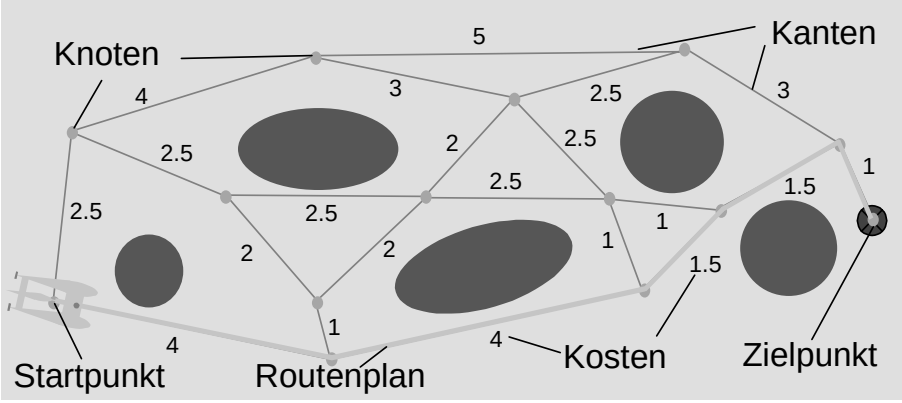


Abbildung 6.48: Geometrischer Graph

Nach Generierung eines solchen Graphen, wird ein Weg (Routenplan) vom Anfangsknoten (Startpunkt) zum Endknoten (Zielpunkt) mit den geringsten Gesamtkosten durch Suchalgorithmen (Dijkstra- [Dij59], A*- [Nil71], D*- Algorithmus [Ste95]) ermittelt. Sie durchsuchen den Graphen, um eine Kombination von Kanten zu ermitteln, welche den Startknoten und den Endknoten mit der geringsten Summe an Kosten verbinden. In den nächsten beiden Abschnitten werden die Arbeitsschritte zur Generierung eines Graphen (Erzeugung der Kanten und Knoten unter Echtzeitanforderungen: Abschnitt 6.2.5.1.1, Ein neuartiges Verfahren zur Bestimmung der Kosten unter Einbeziehung der Strömungsinformation: Abschnitt 6.2.5.1.2) beschrieben.

6.2.5.1.1 Darstellungsmöglichkeiten der Umwelt in einem Graphen

In diesem Abschnitt werden zwei Methoden zur Erzeugung eines geometrischen Graphen (Sichtbarkeitsgraph: Abschnitt 6.2.5.1.1.1, Quadtree- / Octreegraph: Abschnitt 6.2.5.1.1.2) vorgestellt. Die Auswahl dieser Methoden erfolgte auf Grundlage der vorgegebenen Hindernisform (elliptischer Zylinder) und der Forderung nach einer echtzeitfähigen Abarbeitung der Algorithmen unter den gegebenen rechentechnischen Restriktionen des Fahrzeuges.

6.2.5.1.1.1 Sichtbarkeitsgraph

Beim Sichtbarkeitsgraphen bilden die Ecken der Hindernisse die Knoten des Graphen [DJ00]. Hindernisse ohne Ecken (Kreis, Ellipse) werden durch eine Polygonform nachgebildet, welche das Hindernis vollständig einschließt. Alle möglichen Linien zwischen den Knoten, die keine Hindernisse schneiden, werden zu Kanten des Graphen. Abbildung 6.49 zeigt einen so generierten Graphen mit der im Projekt verwendeten elliptischen Hindernisform. Ist das Operationsgebiet sehr groß, wird mit einem „Sektorisierten Sichtbarkeitsgraphen“ gearbeitet [Jac04]. Hier werden nur die Linien zwischen den Knoten von benachbarten Sektoren auf Schnitt mit den Hindernissen überprüft und in den Graphen

eingebunden. Diese Vorgehensweise spart erheblichen Rechenaufwand, senkt die Datenumfang des Graphen und führt zu keiner nennenswerten Verschlechterung der generierten Wege gegenüber einem vollständigen Sichtbarkeitsgraphen.

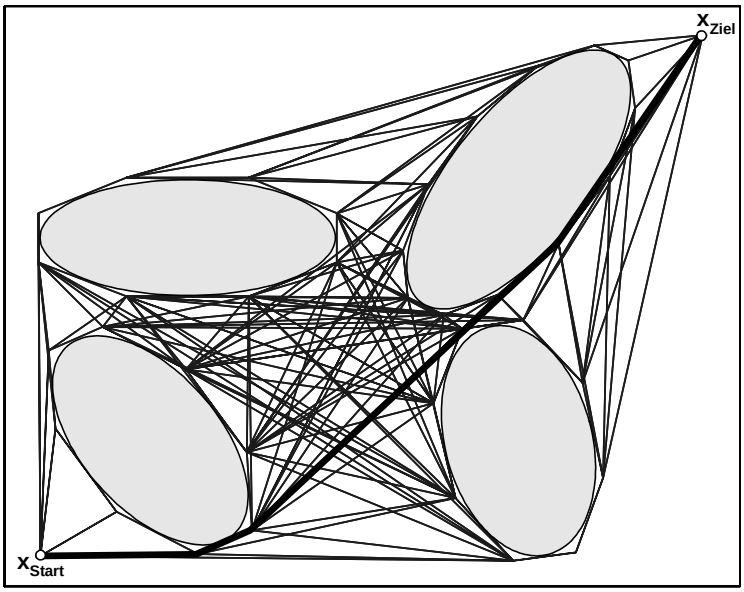


Abbildung 6.49: Sichtbarkeitsgraph

6.2.5.1.1.2 Quadtree- / Octreegraph

Bei diesem Verfahren wird das Operationsgebiet in quadratische (2D-Raum) bzw. kubische (3D-Raum) Sektoren unterteilt. In Abhängigkeit von darin vorhandenen Hindernissen können sie den Zustand „frei“, „belegt“ oder „teilweise belegt“ besitzen. Teilweise belegte Sektoren werden rekursiv weiter geteilt, bis eine maximale Teilungstiefe bzw. eine minimale Sektorgröße erreicht wurde. Als „frei“ oder „belegt“ klassifizierte Sektoren werden nicht weiter geteilt. So findet eine systematische Zerlegung des Operationsgebietes, analog der Verästelung eines Baumes, statt. Nach Erzeugung aller Sektoren wird der Graph aufgebaut. Dazu bilden die Mittelpunkte aller hindernisfreien Sektoren die Knoten des Graphen. Die Kanten sind die Verbindungen zwischen benachbarten hindernisfreien Sektoren (siehe Abbildung 6.50).

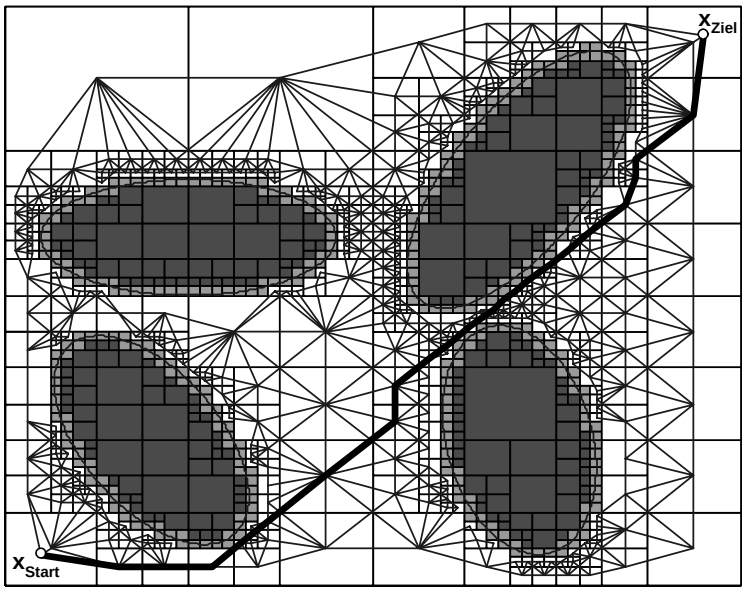


Abbildung 6.50: Quadtreegraph

6.2.5.1.2 Bestimmung des Kostenwertes

Um die Strömungsinformation mit in die Bewertung der Kante einfließen zu lassen, wird als Kriterium die benötigte Zeit t_{Bahn} zum Abfahren der Kante verwendet. Diese Zeit bestimmt sich für die i-te Kante durch Bildung des Quotienten aus der Weglänge s_{Bahn} und der Geschwindigkeit v_{Bahn_ef}, welche das Fahrzeug auf diesem Wegabschnitt gegenüber einem erdfesten Koordinatensystem fährt. Es gilt:

$$t^i_{Bahn} = \frac{s^i_{Bahn}}{v^i_{Bahn\_ef}}. \tag{6.6}$$

Diese Geschwindigkeit v_{Bahn_ef} ist abhängig von der Fahrzeuggeschwindigkeit durchs Wasser v_{Fahr_kf} (Marschgeschwindigkeit), dem Betrag und der Richtung des Strömungsvektors $\mathbf{v}_{Strömung}$ sowie der Richtung des Wegabschnittes $\mathbf{x}_{Bahn\_dir}$. Sie kann durch eine geometrische Schnittberechnung zwischen einer Linie und einem Kreis (2D) bzw. Kugel (3D) [SE03] auf der Basis von Abbildung 6.51 nach den folgenden Beziehungen ermittelt werden:

$$\begin{aligned} &Linie\text{:}\ \ \mathbf{x}\left(v_{Bahn\_ef}\right) = v_{Bahn\_ef}\,\mathbf{x}^{\mathbf{unit}}_{\mathbf{Bahn\_dir}} \\ &Kreis\,/\,Kugel\text{:}\ \ v^2_{Fahr\_kf} = \left\|\mathbf{x} - \mathbf{v}_{\mathbf{Strömung}}\right\|^2 \end{aligned} \tag{6.7}$$

$$\begin{aligned} &disc = \left(\mathbf{x}^{\mathbf{unit}}_{\mathbf{Bahn\_dir}}{}^{\mathbf{T}} \cdot \mathbf{v}_{\mathbf{Strömung}}\right)^2 + v^2_{Fahr\_kf} - \mathbf{v}_{\mathbf{Strömung}}{}^{\mathbf{T}} \cdot \mathbf{v}_{\mathbf{Strömung}} \\ &\text{für } disc > 0\text{:} \\ &v_{Bahn\_ef} = \mathbf{x}^{unit}_{Bahn\_dir}{}^{\mathbf{T}} \cdot \mathbf{v}_{Strömung} + \sqrt{disc} \quad . \end{aligned} \tag{6.8}$$

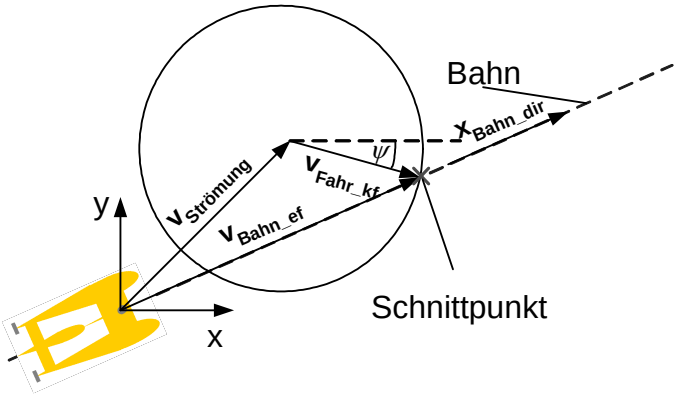


Abbildung 6.51: Definition der Geschwindigkeitsvektoren

Wird die Diskriminante negativ, bedeutet dies, dass das Fahrzeug nicht mehr auf dem Wegabschnitt gehalten werden kann (siehe Abbildung 6.52). Ist die Geschwindigkeit v_{Bahn_ef} negativ, hält sich das Fahrzeug noch auf dem Wegabschnitt, fährt aber rückwärts (siehe Abbildung 6.53). Beide Fälle müssen, durch Setzen eines großen Zahlenwertes für das Kantengewicht, Berücksichtigung finden. So werden solche Wegabschnitte bei der Suche ausgeschlossen, und es kommt nicht dazu, dass ein Fahrzeug in eine starke Gegenströmung gerät und dadurch die Sollroute verlässt.

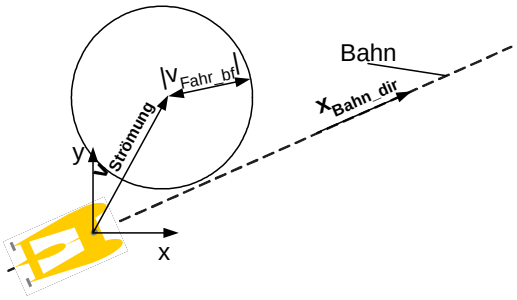


Abbildung 6.52: Negative Diskriminate

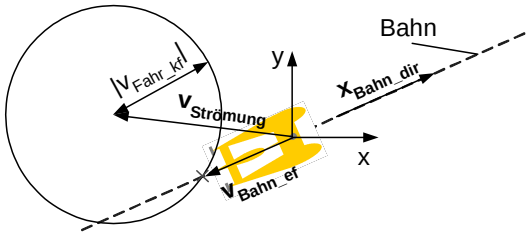


Abbildung 6.53: v_{Bahn_ef} **ist negativ**

Bei Verwendung der oben beschriebenen Kostenfunktion ist es auch möglich, bei komplexen Strömungsprofilen einen zeitoptimalen Weg zu ermitteln. Dies soll an dem nachfolgenden Beispiel erläutert werden.

Die Aufgabe besteht in der Ermittlung eines zeitoptimalen Weges bei der Durchquerung eines Flusslaufes, welcher das nachfolgende Strömungsprofil besitzt:

$$v^x_{Strömung} = 0.0$$

$$v^y_{Strömung} = \frac{4}{b^2} x(b-x) v^0_{Strömung} \qquad (6.9)$$

$$\text{mit } b = 300\text{m} \quad v^0_{Strömung} = 1.8\frac{\text{m}}{\text{s}} \quad .$$

Abbildung 6.54 zeigt die ermittelten Wegverläufe bei Verwendung der Graphenmethode sowie die korrekte zeitoptimale Lösung durch Optimale Steuerung [BH69], [Pap91]. Beide Wegverläufe besitzen ähnliche Profile, welche durch jeden Schwimmer, der einen Fluss mit starker Strömung durchschwommen hat, bestätigt werden. Zuerst schwimmt man gegen die Strömung, um sich dann in der Mitte des Flusses treiben zu lassen und in Richtung des andern Ufers zu schwimmen.

Der Graph wurde bei dieser Aufgabe durch ein Gitternetz aus Knoten aufgebaut. Eine feinere Diskretisierung dieses Netzes würde zu einer weiteren Annäherung an die exakte Lösung führen.

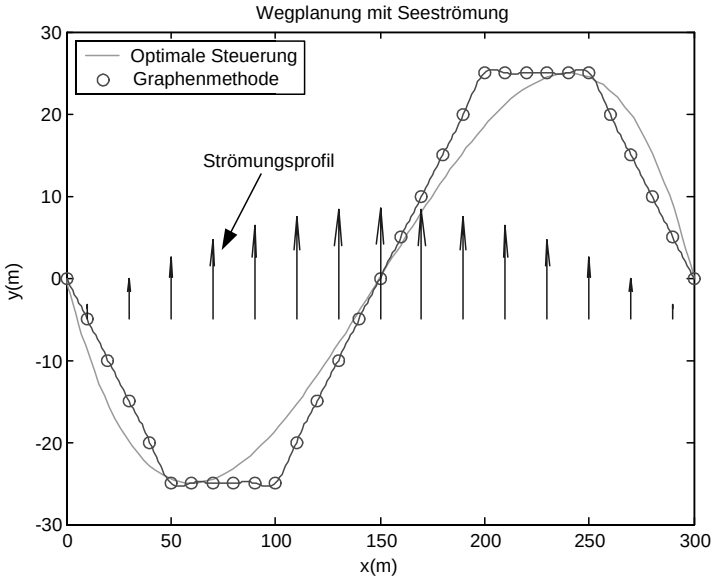


Abbildung 6.54: Wegplanung mit Strömung

6.2.5.1.3 Praktische Umsetzung und Tests

Die durchgeführten Tests erfolgten auf einem Pentium III 450 MHz Desktop. Dieser Rechner entspricht hinsichtlich seiner Rechenleistung der im Fahrzeug eingesetzten Recheneinheit eines Single Board Computers des Typs VP7 mit einem Pentium III Prozessor 450 MHz der Firma SBS Technologies [Sps04].

Für die Tests wurde ein Hindernisparcours mit den Abmaßen von 170 m Länge, 84 m Breite, und 50 m Höhe definiert. Innerhalb dieses Operationsgebietes wurden Hindernisse nach dem in Abbildung 6.55 angegebenen Schema platziert.

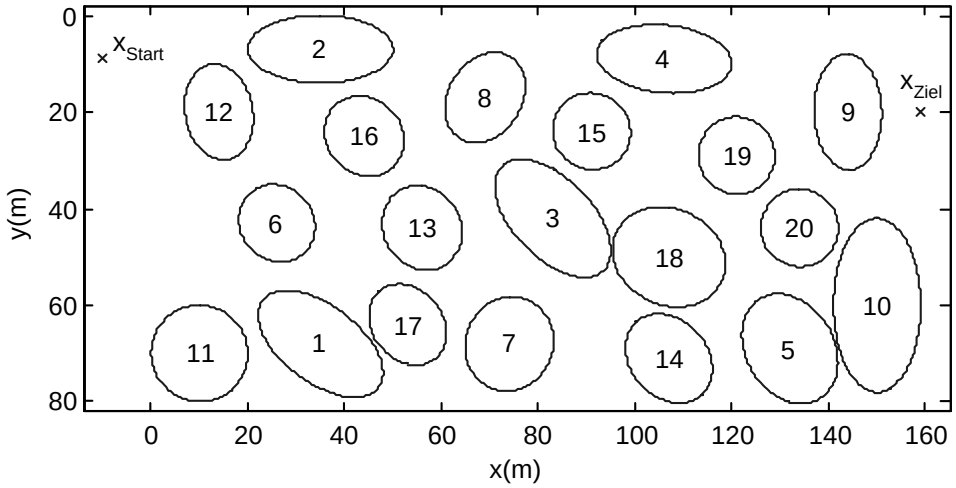


Abbildung 6.55: Hindernisparcours

Die Konturen in Abbildung 6.55 entsprechen den um einen Sicherheitsbereich vergrößerten Hindernissen. Eine Praxisrelevanz von Test15 und Test20 ist durch das begrenzte Auffassvermögen eines Sonars nicht möglich. Diese beiden Tests sollen die Leistungsfähigkeit der Algorithmen überprüfen.

Tabelle 6.7: Testfälle im Hindernisparcours

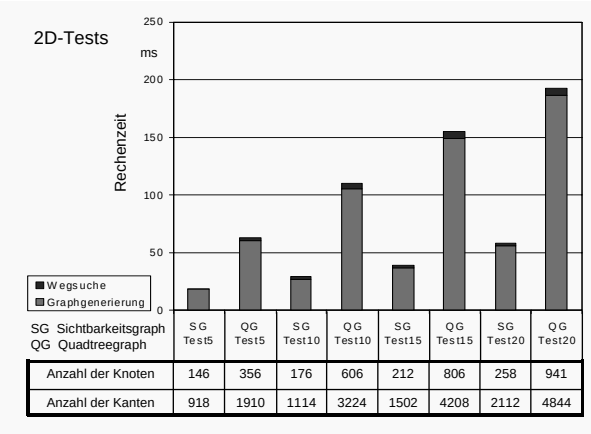
Test	Anzahl der Hindernisse	Hindernisnummern
Test5	5	1-5
Test10	10	1-10
Test15	15	1-15
Test20	20	1-20

Die Testreihen wurden für den 2D und 3D Raum durchgeführt. Bei den 2D – Tests wurden die Hindernisse auf die x-y Ebene projektiert. Die bei den Tests erstellten Graphen beinhalten die in Abschnitt 6.2.5.1.2 vorgestellte Kostenfunktion. Dies führt zur Generierung eines gerichteten Graphen, da jede Kante in Abhängigkeit der Fahrtrichtung einen eigenen Kostenwert beinhaltet.

Für die Pfadsuche wurde die Boost Graph Library [SLL02], [Boo04] eingesetzt. Sie beinhaltet eine Bibliothek mit fertigen Datenstrukturen für Graphen und Suchalgorithmen und unterstützt eine Vielzahl von C++ Compilern. Als Wegsuchalgorithmus kam das Dijkstra-Verfahren zum Einsatz.

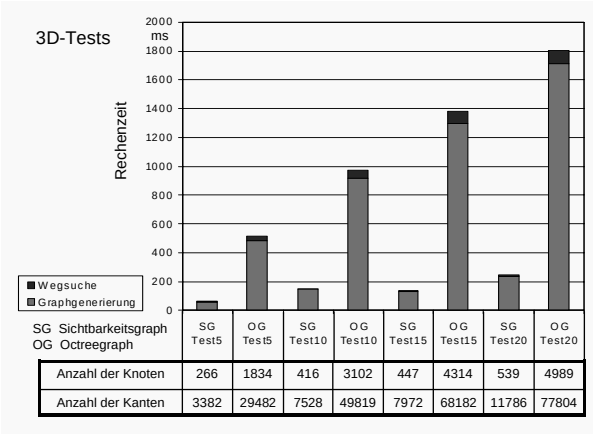
Des Weiteren wurden für die Tests die Compiler Borland C++ Builder Version 6.0 [Bor04], Microsoft Visual C++ (Version 6.0) und Microsoft Visual C++ .NET (Version 7.0) [MSD04]

eingesetzt. Da im Projekt der Borland C++ Builder verwendet wird, stammen die in Abbildung 6.56 und Abbildung 6.57 aufgeführten Rechenzeiten aus den Programmen, welche mit diesem Compiler übersetzt wurden.



SG Sichtbarkeitsgraph QG Quadtreegraph	SG Test5	QG Test5	SG Test10	QG Test10	SG Test15	QG Test15	SG Test20	QG Test20
Anzahl der Knoten	146	356	176	606	212	806	258	941
Anzahl der Kanten	918	1910	1114	3224	1502	4208	2112	4844

Abbildung 6.56: 2D Tests



SG Sichtbarkeitsgraph OG Octreegraph	SG Test5	OG Test5	SG Test10	OG Test10	SG Test15	OG Test15	SG Test20	OG Test20
Anzahl der Knoten	266	1834	416	3102	447	4314	539	4989
Anzahl der Kanten	3382	29482	7528	49819	7972	68182	11786	77804

Abbildung 6.57: 3D Tests

Der Vergleich von Abbildung 6.56 und Abbildung 6.57 zeigt ein deutliches Anwachsen der Rechenzeit bei den 3D Tests. Der größte Zeitaufwand steckt in der Generierung des Graphen. Die Zeit für die Wegsuche nimmt dabei einen verschwindend kleinen Anteil ein. Der Rechenaufwand zur Generierung eines Quadtree- / Octreegraphen liegt um das 3fache im 2D Raum und um das 7fache im 3D Raum höher als beim Sichtbarkeitsgraphen. Dies ist durch die notwendige minimale Sektorgrösse (1m x 1m x 6m) und der damit verbundenen zeitaufwendigen Generierung der Datenstrukturen für den 2D bzw. 3D-Baum [Kle97] und des Graphen begründet. Nur durch eine solch feine Diskretisierung können auch „freie“ Sektoren zwischen den Hindernissen generiert werden.

Der Einsatz der Microsoft-Compiler brachte im Durchschnitt eine Senkung der Rechenzeit um 35%. Für das Projekt wurde der Sichtbarkeitsgraph favorisiert. Dies ist durch seinen geringeren Rechenaufwand und durch die Form der ermittelten Wege (vergleiche Abbildung 6.49 und Abbildung 6.50) begründet. Es finden bei diesem Verfahren weniger Kursänderungen als beim Octreegraphen während des Durchfahrens eines Parcours statt.

6.2.5.2 Reaktive Ausweichsteuerung

Die Reaktive Ausweichsteuerung muss bei plötzlich auftauchenden Hindernissen die Wegeplanung ablösen und die Fahrzeugführung übernehmen. Ein anderer Einsatzfall ist bei Bereitstellung „ungenauer“ Objektdaten durch das Modul „Objekterkennung“ gegeben. Unter ungenau ist ein stochastisches Variieren der Abmaße und der Lage der Objekte infolge der begrenzten Reichweite der Aufklärungs-Sensorik (z. B. Forward Looking Sonar) sowie der entfernungsabhängigen Detektionsgüte zu verstehen. In solchen Fällen ist ein Ausweichverfahren gefordert, welches eine hohe Robustheit gegenüber solchen Variationen besitzt. Diese Forderung wird durch die Reaktive Ausweichsteuerung erfüllt, da sie die Steuerkommandos (Sollkurs, -geschwindigkeit) für den Autopiloten nur auf der Basis der aktuellen Lage der Objekte zum Fahrzeug generiert. Eine Wegeplanung, welche eine gewisse Konstanz der Objektdaten zwischen Erstellung und Abarbeitung des Planes voraussetzt, wird nicht durchgeführt.

Die Idee des in diesem Abschnitt vorgestellten neuen Verfahrens besteht in der Verwendung von Bahnlinien, welche von jeder beliebigen Position des Operationsgebietes einen Weg zum Zielpunkt aufzeigen. Der Sollkurs des Fahrzeuges wird durch Bildung des Bahnanstiegs (Gradienten G) an der aktuellen Fahrzeugposition bestimmt. So kann das Fahrzeug von seiner aktuellen Position aus die aktuelle Bahnlinie bzw. ihren Gradienten verwenden, um vorbei an den Hindernissen zum Zielpunkt geführt zu werden. Da für die reaktive Ebene nur

die x-y Ausdehnung der Hindernisobjekte berücksichtigt wird, soll in diesem Abschnitt auch nur der 2D-Raum behandelt werden. Es ist jedoch auch möglich, das vorgestellte Verfahren für den 3D-Raum durch entsprechende Modifikationen zu verwenden.

Das in diesem Abschnitt beschriebene Verfahrens basiert auf den Arbeiten von [Gul95], [Gul97]. Es werden Gradientenlinien durch künstliche harmonische Dipol-Potentiale erzeugt. Durch die Verwendung solcher Dipol-Potentiale können die negativen Eigenschaften der künstlichen Potentialfelder kompensiert werden. Solche Eigenschaften sind das „Festsetzen“ des Fahrzeuges in lokalen Minima, die oszillierende Bewegung in engen Durchfahrten sowie das Sperren von möglichen Routen durch das resultierende abstoßende Kraftfeld am Eingang einer Durchfahrt [Gul97].

In den nächsten Abschnitten wird ein neu entwickeltes Verfahren vorgestellt, welches die Gewinnung der Gradienten aus einer geometrischen Konstruktion der Gradientenlinien ermöglicht. Des Weiteren werden Möglichkeiten aufgezeigt, ein Ausweichen unter Einbeziehung der Geschwindigkeitsinformationen der Seeströmung und der Hindernisse durchzuführen.

6.2.5.2.1 Erzeugung von Gradientenlinien durch harmonische Dipol-Potentiale

Für die nachfolgenden Ausführungen sei angenommen, dass ein Hindernis durch einen Kreis beschrieben wird. Fasst man den Kreis als positive Ladung und den anzufahrenden Zielpunkt als negative Punktladung auf, so bildet diese Struktur einen Dipol, dessen Feldlinien von der positiven zur negativen Ladung führen. Der Wert für die negative Ladung im Zielpunkt wird zu –1 gewählt. Die positive Ladung des Hindernisses wird durch die Gleichung:

$$q = \frac{R}{R + D} \tag{6.10}$$

beschrieben. Dabei ist *R* der Radius des Hinderniskreises. *D* beschreibt den Abstand zwischen den beiden Ladungen. Durch Gleichung (6.10) wird garantiert, dass alle Gradientenlinien, welche innerhalb des Kreises beginnen, herausgeführt werden und im Zielpunkt enden. Gradientenlinien, welche außerhalb des Kreises beginnen, verlaufen ausschließlich außerhalb des Kreises. Für die Beweisführung sei auf [Gul95] und [Gul97] verwiesen.

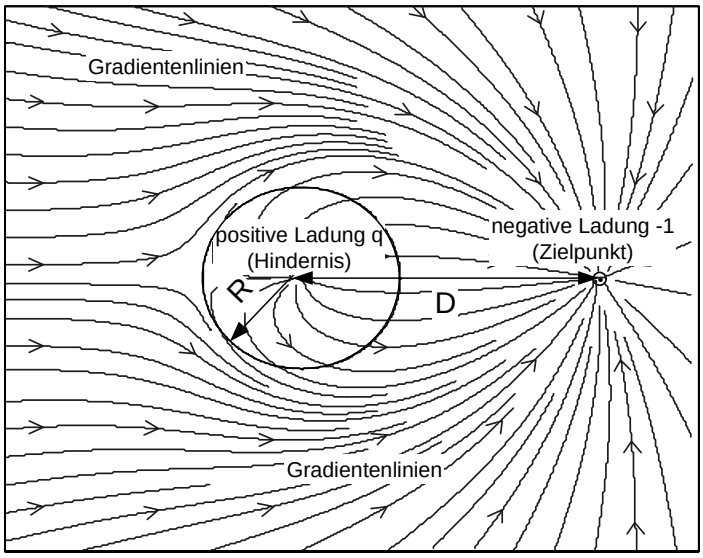


Abbildung 6.58: Gradientenlinien des Dipols

Für die Ermittlung des resultierenden Gradienten **G** an der aktuellen Fahrzeugposition sind die beiden Gradienten für die positive und negative Ladung getrennt zu berechnen und durch eine Superposition zusammenzufassen. Abbildung 6.58 zeigt das so entstandene Potentialfeld mit den dazugehörigen Gradientenlinien.

Da die Gradientenlinien das Hindernis tangieren, aber nicht schneiden, können, muss der Radius R des Kreises um einen solchen Betrag vergrößert werden, welcher eine Kollision zwischen Fahrzeug und Objekt verhindert. Hier bietet sich eine Vergrößerung mit der n-fachen maximalen Ausdehnung des Fahrzeuges an. Der so entstandene Kreis wird auch als Sicherheitszone bezeichnet [Gul97].

Befindet sich das Fahrzeug auf der dem Zielpunkt abgewandten Seite und fährt in der Nähe der Symmetrieachse des Dipols, wird es erst kurz vor Erreichen des Hinderniskreises zum Umfahren des Hindernisses gebracht (siehe Abbildung 6.58). Dabei sind starke Lenkkommandos notwendig, um das Fahrzeug auf der vorgegebenen Gradientenlinie zu halten. Des Weiteren entspricht die Form der Gradientenlinien in diesem Bereich und an den seitlichen Vorbeiführungen nicht einer wegoptimalen Fahrtroute. Diese Effekte nehmen mit größer werdenden Parametern von R und D zu. Sie sollen durch die geometrische Konstruktion von Gradientenlinien zur wegoptimalen Führung im neuen Verfahren beseitigt werden.

6.2.5.2.2 Erzeugung von Gradientenlinien durch geometrische Konstruktion

Einführend sei an dieser Stelle erwähnt, dass die in diesem Abschnitt erzeugten Gradientenlinien keine physikalischen „Vorbilder“ haben. Sie sollen lediglich zu einer wegoptimalen Fahrtroute führen.
Hierzu wird das Operationsgebiet in einzelne Sektoren aufgeteilt (siehe Abbildung 6.59). Die Einteilung der Sektoren erfolgt in Abhängigkeit der Fahrzeugposition $\mathbf{x}_{\mathbf{Fahr}}$ vom Hinderniszentrum $\mathbf{x}_{\mathbf{Hind}}$ und vom Zielpunkt.

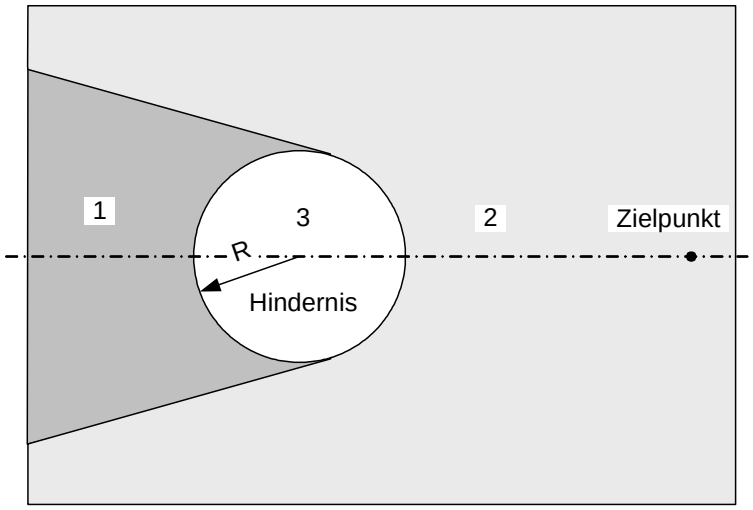


Abbildung 6.59: Einteilung des Operationsgebietes in Sektoren

Wenn die Bedingung ($|\mathbf{x}_{\mathbf{Fahr}} - \mathbf{x}_{\mathbf{Hind}}| < R$) erfüllt ist, befindet sich das Fahrzeug im Inneren des Sicherheitskreises. Es ist der Sektor 3 aktiv. Der Gradient wird dabei durch die Gleichung:

$$\mathbf{x}_{\mathbf{Hind}}^{\mathbf{Distanz}} = \mathbf{x}_{\mathbf{Fahr}} - \mathbf{x}_{\mathbf{Hind}}$$
$$\mathbf{G} = \frac{\mathbf{x}_{\mathbf{Hind}}^{\mathbf{Distanz}}}{\left|\mathbf{x}_{\mathbf{Hind}}^{\mathbf{Distanz}}\right|} \quad . \tag{6.11}$$

berechnet. Die Aktivierung des Sektors 3 scheint im ersten Moment ausgeschlossen. An dieser Stelle sei jedoch noch einmal festgestellt, dass der Radius R eine Sicherheitszone beschreibt, in welcher sich das Hindernis befindet. So ist es durchaus möglich, dass ein Fahrzeug in diesen Sicherheitskreis eindringen kann (starke Strömung, zu späte Erfassung des Hindernisses). In einem solchen Fall muss das Fahrzeug wieder sicher aus dem Kreis geführt werden, was eine Gradientenbeschreibung im Innern des Kreises erfordert.

Wird die Sicht zwischen Fahrzeug und Zielpunkt durch den Kreis verdeckt, ist keine direkte Anfahrt zum Zielpunkt möglich (Sektor 1). In diesem Fall muss der Richtungsvektor der Tangentengerade von $\mathbf{x}_{\mathbf{Fahr}}$ zum Kreis in Abhängigkeit der Lage von $\mathbf{x}_{\mathbf{Fahr}}$ zur Symmetrieachse bestimmt werden. Der Gradient ist dabei der normierte Richtungsvektor $\mathbf{x}_{\mathbf{Tangente}}^{\mathbf{Richtung}}$:

$$\mathbf{G} = \frac{\mathbf{x}_{\mathbf{Tangente}}^{\mathbf{Richtung}}}{\left|\mathbf{x}_{\mathbf{Tangente}}^{\mathbf{Richtung}}\right|}. \tag{6.12}$$

Gibt es eine direkte Verbindung zwischen Fahrzeug und Zielpunkt, ist Sektor 2 aktiv. Dann gilt:

$$\begin{aligned} \mathbf{x}_{\mathbf{Ziel}}^{\mathbf{Distanz}} &= \mathbf{x}_{\mathbf{Ziel}} - \mathbf{x}_{\mathbf{Fahr}} \\ \mathbf{G} &= \frac{\mathbf{x}_{\mathbf{Ziel}}^{\mathbf{Distanz}}}{\left|\mathbf{x}_{\mathbf{Ziel}}^{\mathbf{Distanz}}\right|} \end{aligned} \quad . \tag{6.13}$$

Um beim Austreten der Gradientenlinien aus dem Sicherheitskreis einen kontinuierlichen Übergang zu den Gradientenlinien der Sektoren 1 und 2 zu erreichen, wurde ein Radius R_{slid} eingeführt. In dem dadurch entstehenden Kreisring zwischen R und R_{slid} findet eine gewichtete Überlagerung unter Verwendung des Wichtungsfaktors α zwischen den Gradienten außerhalb und innerhalb des Kreises nach der Beziehung

$$\begin{aligned} \alpha &= \frac{\left|\mathbf{x}_{\mathbf{Fahr}} - \mathbf{x}_{\mathbf{Hind}}\right| - R}{R_{slid} - R} \\ \mathbf{G} &= \alpha \underbrace{\mathbf{G}_{\mathbf{Sektor1,2}}}_{\substack{\text{Außerhalb des} \\ \text{Kreises}}} + \left(1-\alpha\right) \underbrace{\mathbf{G}_{\mathbf{Sektor3}}}_{\substack{\text{Innerhalb des} \\ \text{Kreises}}} \end{aligned} \tag{6.14}$$

statt. Abbildung 6.60 zeigt ein so erzeugtes Gradientenfeld.

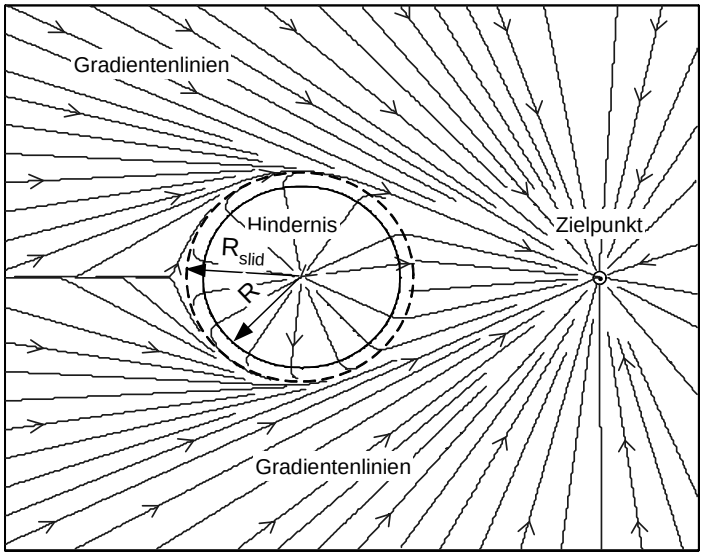


Abbildung 6.60: Gradientenlinien bei geometrischer Konstruktion

6.2.5.2.3 Einbeziehung der Geschwindigkeitsinformationen

Unter realen Umweltbedingungen kann es vorkommen, dass ein Hindernis sich bewegt (beschrieben durch $\mathbf{v}_{\mathbf{Hind}}$) oder eine starke Seeströmung $\mathbf{v}_{\mathbf{Strömung}}$ vorhanden ist. In solchen Fällen ist es sinnvoll, diese Geschwindigkeitsinformationen bei der Generierung der Gradientenlinien mit einfließen zu lassen. So soll es möglich sein, ein Ausweichmanöver schon früher zu beginnen, wenn sich ein Hindernis auf das Fahrzeug zu bewegt oder eine starke Rückenströmung vorhanden ist. In beiden Fällen wird ein mögliches Kollidieren durch unzureichendes Reagieren vermieden. Eine Möglichkeit, dies zu erreichen, ist die Vergrößerung der Sicherheitszone in Richtung einer resultierenden Geschwindigkeit $\mathbf{v}_{\mathbf{res}}$, welche durch die folgende Beziehung beschrieben wird:

$$\mathbf{v}_{\mathbf{res}} = \mathbf{v}_{\mathbf{Hind}} - \mathbf{v}_{\mathbf{Strömung}} . \tag{6.15}$$

Die Sicherheitszone wird damit von ihrer Kreisform mit dem Radius R in eine Ellipse überführt (siehe Abbildung 6.61).

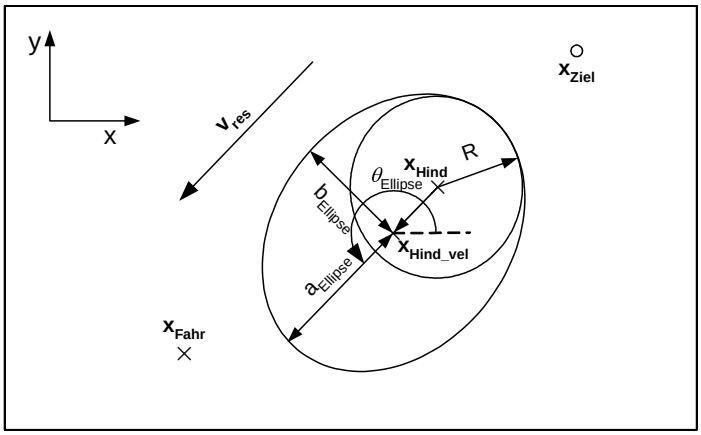


Abbildung 6.61: Verformung der Sicherheitszone

Die geometrischen Maße der so entstandenen Ellipse werden durch die nachfolgenden Gleichungen bestimmt:

$$\begin{aligned} a_{Ellipse} &= R\left(1+v_{rel}\right) \\ b_{Ellipse} &= R\sqrt{\left(1+v_{rel}\right)} \\ \theta_{Ellipse} &= \tan^{-1}\left(\frac{v_{res}^{y}}{v_{res}^{x}}\right) \quad . \end{aligned} \tag{6.16}$$

Der Koeffizient v_{rel} beschreibt das Verhältnis zwischen dem Betrag der resultierenden Geschwindigkeit $|\mathbf{v}_{\mathbf{res}}|$ und der vorgegebenen Marschgeschwindigkeit v_{Marsch} des Fahrzeuges und wird berechnet aus:

$$v_{rel} = \frac{\left|\mathbf{v}_{\mathbf{res}}\right|}{v_{Marsch}} . \tag{6.17}$$

Das Zentrum der Ellipse wird durch den Zusammenhang

$$\mathbf{x}_{\mathbf{Hind\_vel}} = \mathbf{x}_{\mathbf{Hind}} + \begin{bmatrix} \cos\theta_{Ellipse} R\ v_{rel} \\ \sin\theta_{Ellipse} R\ v_{rel} \end{bmatrix} \tag{6.18}$$

beschrieben. Die Berechnung der Gradientenlinien für die entstandene elliptische Form des Sicherheitsgebietes erfolgt auf der Basis der Ausführungen im Abschnitt 6.2.5.2.4.

6.2.5.2.4 Anwendung bei elliptischen Sicherheitszonen

Bei den vorhergehenden Ausführungen wurde vorausgesetzt, dass ein Hindernis durch einen Kreis beschrieben wird. Da durch die Kreisform viele Objekte ungünstig beschrieben werden, ist eine elliptische Form vorteilhafter. Die Anpassung des vorgestellten Verfahrens an eine elliptische Form wäre allerdings sehr rechenaufwendig und bei Einbeziehung der Geschwindigkeitsinformation (Abschnitt 6.2.5.2.3) für nichtkreisförmige Hindernisse gar nicht realisierbar. Deshalb wird eine Transformation durchgeführt, welche eine gedrehte Ellipse mit der Position $\mathbf{x}_{\mathbf{Ellipse}}$ in einen im Koordinatenursprung liegenden Kreis überführt. Die Berechnung der Gradientenlinien erfolgt dann für kreisförmige Hindernisse, wie in Abschnitt 6.2.5.2.2 beschrieben und anschließender Rücktransformation. Die Transformation für einen beliebigen Punkt **x** in diesem Bildbereich setzt sich so aus mehreren Teilschritten zusammen (siehe Abbildung 6.62):

1) Translation des Punktes um die Position $\mathbf{x}_{\mathbf{Ellipse}}$. (Verschiebung der Ellipse in den Koordiantenursprung des Bildbereiches.)

$$\mathbf{x}_{\mathbf{trans}} = \mathbf{x} - \mathbf{x}_{\mathbf{Ellipse}} \tag{6.19}$$

2) Rotation des Punktes um den Rotationswinkel der Ellipse $\theta_{Ellipse}$.
(Rotation der Ellipse zur Erreichung einer achsenparallelen Lage der Ellipse im Bildbereich.)

$$\mathbf{x}_{\mathbf{trans\_rot}} = \begin{bmatrix} \cos\theta_{Ellipse} & \sin\theta_{Ellipse} \\ -\sin\theta_{Ellipse} & \cos\theta_{Ellipse} \end{bmatrix} \mathbf{x}_{\mathbf{trans}} \tag{6.20}$$

3) Skalierung des Punktes mit den inversen Werten der halben Haupt- und Nebenachse ($a_{Ellipse}$ und $b_{Ellipse}$) der Ellipse. (Erreichung einer Kreisform für die Ellipse mit R = 1.)

$$\mathbf{x}_{\mathbf{0}} = \begin{bmatrix} \frac{1}{a_{Ellipse}} & 0 \\ 0 & \frac{1}{b_{Ellipse}} \end{bmatrix} \mathbf{x}_{\mathbf{trans\_rot}}. \tag{6.21}$$

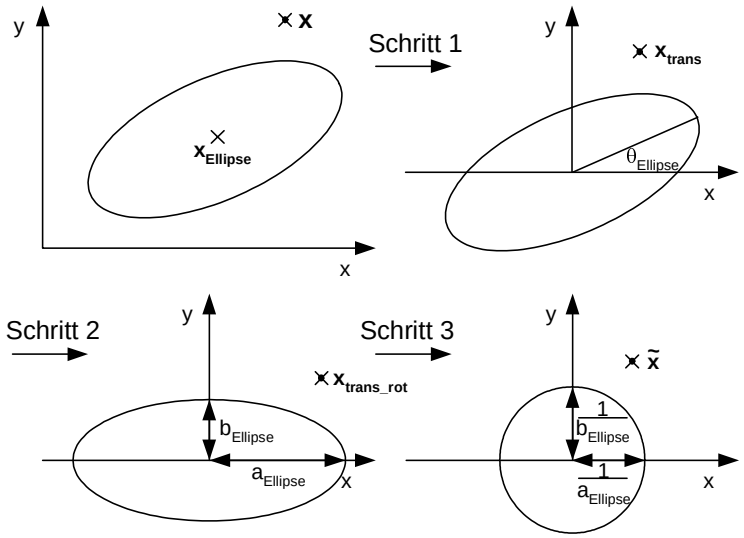


Abbildung 6.62: Schritte der Transformation

Nach Durchführung dieser Transformation können die eigentlichen Berechnungen im Bildbereich durchgeführt werden. Der hierbei berechnete Gradient $\mathbf{G}$ muss anschließend wieder vom Bildbereich in den Originalbereich rücktransformiert werden (Gleichung (6.22), (6.23)). Es gilt:

Rückskalierung:

$$\mathbf{G}_{\mathbf{rescal}} = \begin{bmatrix} a_{Ellipse} & 0 \\ 0 & b_{Ellipse} \end{bmatrix} \mathbf{G} \tag{6.22}$$

Rückrotation:

$$\mathbf{G} = \begin{bmatrix} \cos\theta_{Ellipse} & -\sin\theta_{Ellipse} \\ \sin\theta_{Ellipse} & \cos\theta_{Ellipse} \end{bmatrix} \mathbf{G}_{\mathbf{rescal}} . \tag{6.23}$$

6.2.5.2.5 Anwendung für mehrere Hindernisse

Bei den bisherigen Ausführungen wurde nur ein einzelnes Hindernis im Operationsgebiet betrachtet. Sind mehrere vorhanden, muss darauf geachtet werden, dass sich die Sicherheitsbereiche der einzelnen Hindernisse nicht überschneiden. Lässt sich dies nicht vermeiden, müssen die entsprechenden Hindernisse durch eine gemeinsame Sicherheitszone abgedeckt werden [Gul97].

Bei der Bestimmung des Gradienten werden nur die Gradientenlinien des dem Fahrzeug nächstgelegenen Hindernisses verwendet. Dies führt dazu, dass bei der Durchfahrung eines Hindernisparcours ein Umschalten zum jeweils aktuellen Gradientenlinienfeld erfolgt. Um beim Übergang zwischen zwei Gradientenfeldern einen homogenen Verlauf zu erzielen, ist es notwendig, statt des binären Umschaltens ein „kontinuierliches“ Umschalten durchzuführen. In [Gul97] wurde ein Bereich entlang der Grenzlinie zweier benachbarter Gradientenlinienfelder definiert (d_{Linie_max}), in welchem sich die Gradienten der einzelnen Felder mittels eines Wichtungsfaktors α linear überlagern. Dieser Wichtungsfaktor wurde proportional zum Fahrzeugabstand von der Grenzlinie gewählt. Eine vereinfachte

Bestimmung dieses Abstandes kann durch Gleichung (6.24) auf Grundlage von Abbildung 6.63 erfolgen. Es gilt:

$$d_{Linie} = \frac{d_{Fahr\_n}^2 - d_{Fahr\_m}^2}{2d_{m\_n}}. \tag{6.24}$$

Die Abstände d_{Fahr_n} und d_{Fahr_m} beschreiben die kürzeste Entfernung von der aktuellen Fahrzeugposition zum jeweiligen Hindernis.

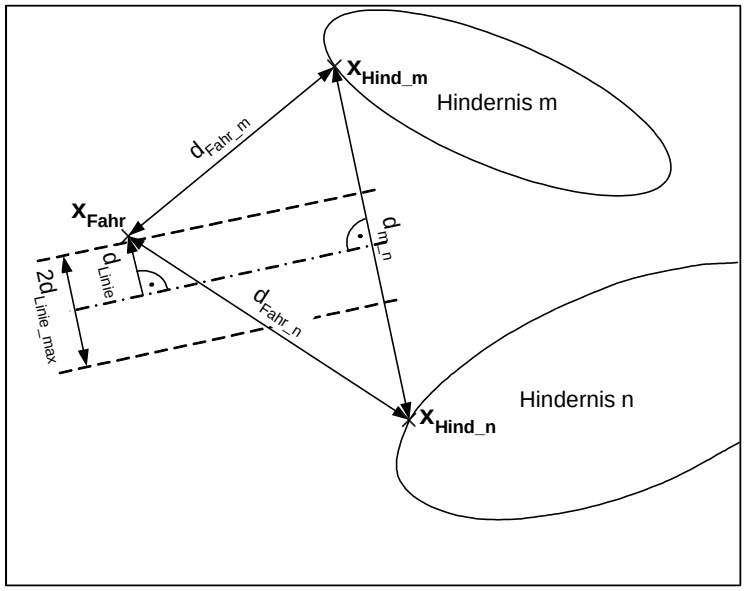


Abbildung 6.63: Vereinfachte Bestimmung von d_{Linie}

Der resultierende Gradient bestimmt sich so durch die folgenden Beziehungen:

$$\mathbf{G} = \begin{cases} \mathbf{G}_{\mathbf{Hind\_m}} & \text{für } d_{Linie} > d_{Linie\_max} \\ \mathbf{G}_{\mathbf{Hind\_n}} & \text{für } d_{Linie} < -d_{Linie\_max} \\ \alpha\mathbf{G}_{\mathbf{Hind\_m}} + (1-\alpha)\mathbf{G}_{\mathbf{Hind\_n}} & \text{für } |d_{Linie}| < d_{Linie\_max} \end{cases} \tag{6.25}$$

mit dem Wichtungsfaktor α

$$\alpha = 0.5\left(1 + \frac{d_{Linie}}{d_{Linie\_max}}\right). \tag{6.26}$$

6.2.5.2.6 Überschneidende Sicherheitsbereiche

Kommt es zu Überschneidungen zwischen zwei oder mehreren Sicherheitszonen, müssen diese Gebiete durch eine gemeinsame Sicherheitszone abgedeckt werden. Diese Zone sollte eine Ellipsenform besitzen, welche die einzelnen Sicherheitszonen vollständig einschließt aber möglichst wenig befahrbares Gebiet beinhaltet. Zur Lösung dieses Problems

wurde der Ansatz zur „Berechnung kleinster Ellipsoide um Punktmengen“ verwendet [GS98]. Dazu wurde um jede Sicherheitszone ein Polygon definiert, welches das Hindernis einschließt. Um die Stützstellen aller so gebildeten Polygone wird eine Ellipse unter Verwendung der GCAL Bibliothek [CGAL04] berechnet.

Die Berechnung der Polygonstützstellen erfolgte auf der Basis von normierten Stützstellen um einen Einheitskreis. Diese werden durch Skalierung und Rotation der entsprechenden Ellipsenform angepasst (siehe Gleichung (6.22) und (6.23)). Die Stützstellen $\mathbf{S}_{0\ldots n-1}$ werden für den ersten Quadranten des Einheitskreises durch Schnittpunktberechnung nach Gleichung (6.28) unter Verwendung der in Gleichung (6.27) definierten Strahlen berechnet und für die anderen Quadranten durch entsprechenden Vorzeichenwechsel angepasst (siehe Abbildung 6.64).

$$\mathbf{P}_n = \begin{bmatrix} \cos(n \cdot \alpha) \\ \sin(n \cdot \alpha) \end{bmatrix} \quad \mathbf{d}_n = \begin{bmatrix} -\sin(n \cdot \alpha) \\ \cos(n \cdot \alpha) \end{bmatrix} \tag{6.27}$$

$$\mathbf{E} = \mathbf{P}_{n+1} - \mathbf{P}_n \quad t = \frac{\mathbf{E} \times \mathbf{d}_{n+1}}{\mathbf{d}_n \times \mathbf{d}_{n+1}}$$
$$\mathbf{S}_n = \mathbf{P}_n + t \cdot \mathbf{d}_n \tag{6.28}$$

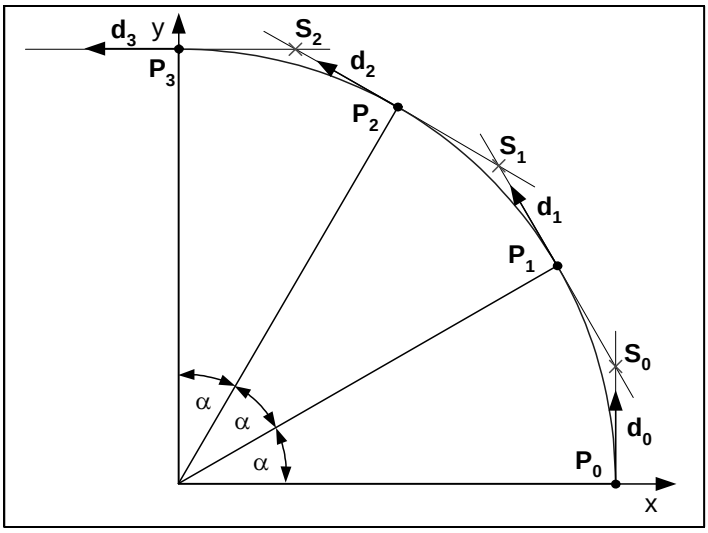


Abbildung 6.64: Ermittlung der Stützstellen auf einen Einheitskreis

Abbildung 6.65 zeigt die ermittelten Stützstellen der einzelnen Sicherheitszonen und die daraus berechnete Ellipse. In der Vergrößerung ist deutlich zu erkennen, dass die Polygonkanten die Sicherheitszone tangieren aber nicht schneiden.

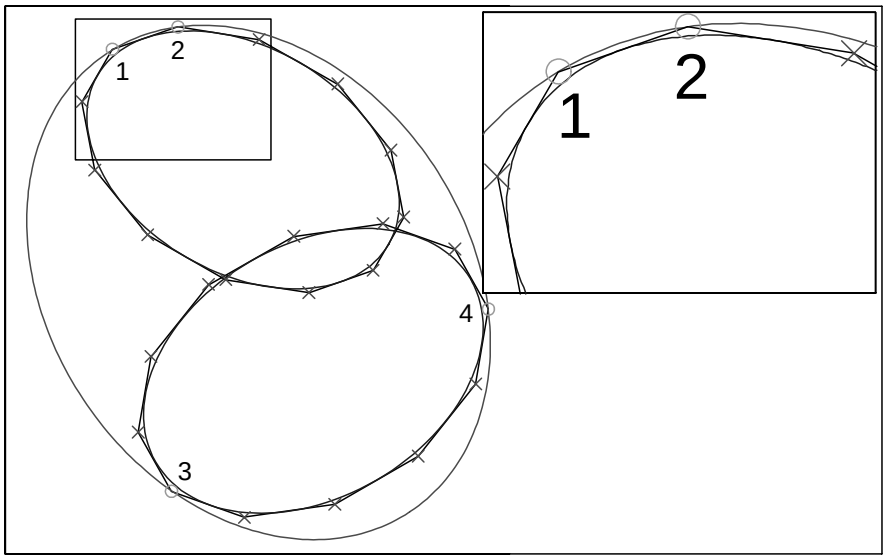


Abbildung 6.65: Kleinste eingeschlossene Ellipse

Die Gradientenberechnung im Innern der Ellipse erfolgt in Analogie zum Abschnitt 6.2.5.2.5. Befindet sich die Fahrzeugposition im Inneren einer Sicherheitszone, wird als nächste Position $\mathbf{x}_{obst\_m}$ zum Fahrzeug die Fahrzeugposition $\mathbf{x}_{veh}$ definiert. Abbildung 6.66 zeigt das so gebildete Gradientenfeld.

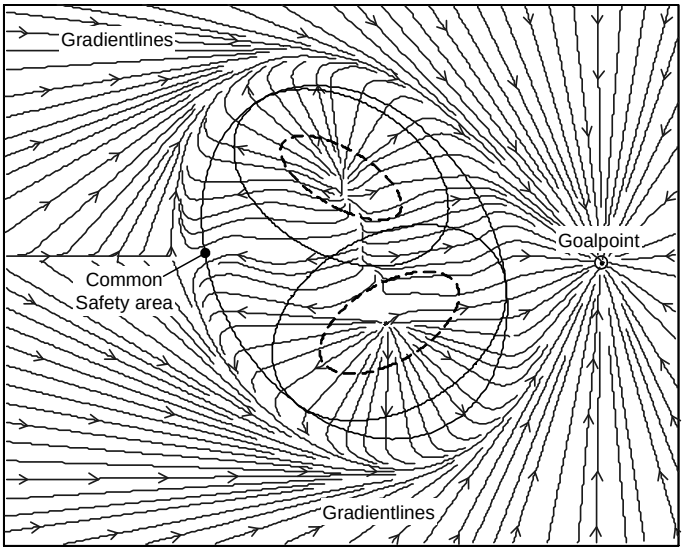


Abbildung 6.66: Gradientenlinien bei zusammengefassten Sicherheitszonen

6.2.5.2.7 Test des Verfahrens

Abschließend soll das neu entwickelte Verfahren der geometrischen Konstruktion mit dem Verfahren der harmonischen Dipolpotentiale verglichen werden. Dazu wurde ein Hindernisparcours, der in Abbildung 6.68 dargestellt ist, definiert. Das Fahrzeug soll mit einer Marschgeschwindigkeit von 2m/s durch diesen Parcours fahren. Es zeigte sich, dass bei Verwendung des geometrischen Verfahrens die Sollkursrate r wesentlich geringere Maximalwerte als das Verfahren der harmonischen Dipolpotentiale besitzt (siehe Abbildung 6.67). Dies ist für den praktischen Einsatz ein wichtiges Kriterium, da das eingesetzte Fahrzeug schnellen Sollkursänderungen nicht folgen kann, was zum Abweichen von der Gradientenlinie und damit zu einer möglichen Kollision führt. Des Weiteren bewirkt jede

schnelle Sollkursänderung eine Vergrößerung des Navigationsfehlers. Schon aus diesem Grund sollten solche Manöver vermieden werden. Die benötigte Weglänge zum Durchfahren des Parcours beträgt beim Geometrischen Verfahren bis zu 9% weniger als bei der Methode der harmonischen Dipolpotentiale. Die Trajektorienverläufe des geometrischen Verfahrens sind der Form von optimalen Wegen, wie sie in [WHI99] unter Verwendung einer Evolutionsstrategie generiert wurden, sehr ähnlich.

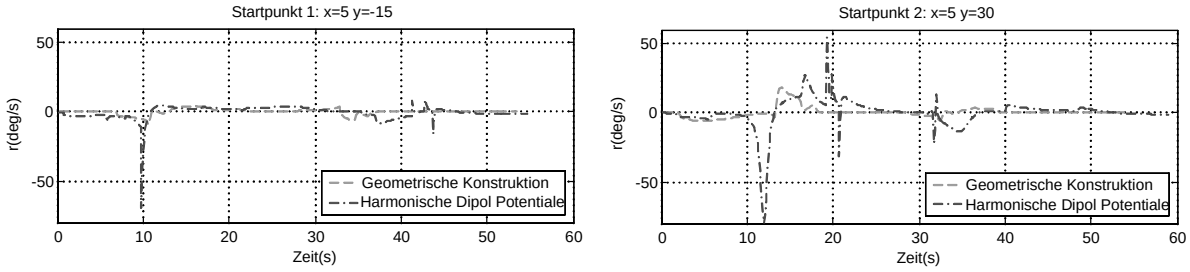


Abbildung 6.67: Auswertung der Sollkursraten

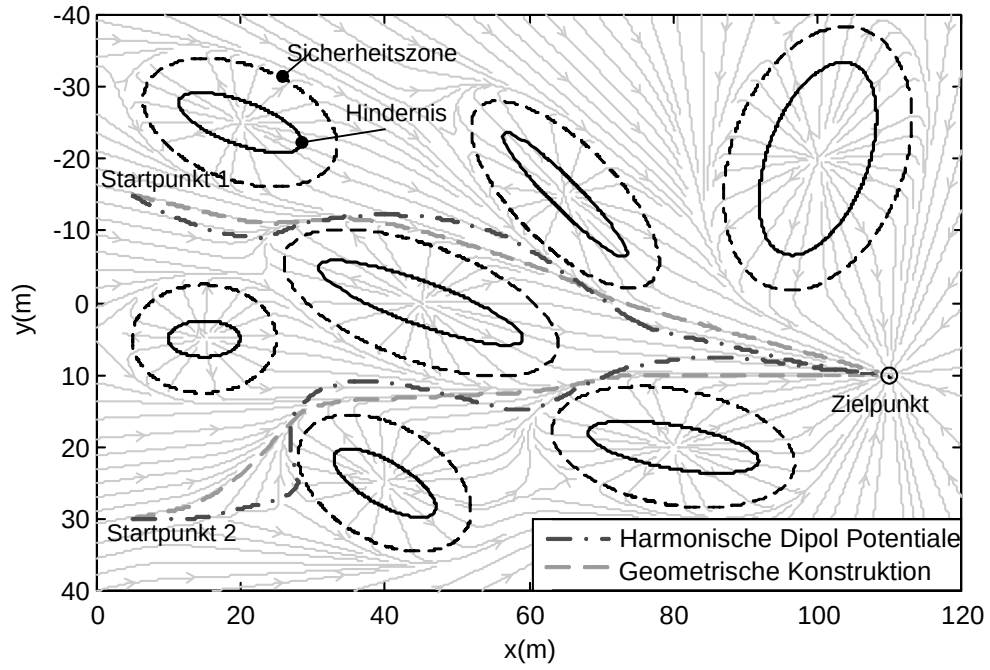


Abbildung 6.68: Wege durch einen Hindernisparcours

6.2.6 Identifikationssystem

Bei der Durchführung einer Mission kann eine geforderte Aufgabe die Identifikation von Objekten sein. Eine Identifikation kann eine visuelle oder kartographische Erfassung des Objektes und/oder eine Bestimmung seiner physikalischen/chemischen Eigenschaften beinhalten. Die Führung des Fahrzeuges wird dazu an das Submodul *Identifikationssystem* übergeben. Hier werden, nach Vorgaben über die aktuelle Objektgeometrie und den Anforderungen des Fahrzeuges während der Identifikation (Objektabstand, Ausrichtung zum Objekt, ggf. Fahrgeschwindigkeit um das Objekt), die Sollwerte für den unterlagerten Autopiloten gebildet (siehe Abbildung 6.69). In Abhängigkeit der Objektgrösse werden die zwei Arbeitsmodies *Ausrichten auf ein Objekt* und *Umfahren eines Objektes* unterschieden. Sie werden in den nachfolgenden Abschnitten kurz vorgestellt.

Nach Abschluss eine Identifikation wird ein anzufahrender Zielpunkt auf der aktuellen Missionsroute durch das Submodul *Zielpunktgenerierung* (siehe Abbildung 6.37) ermittelt. Dieser Zielpunkt wird dann durch das aktivierte Hindernisvermeidungssystem angefahren.

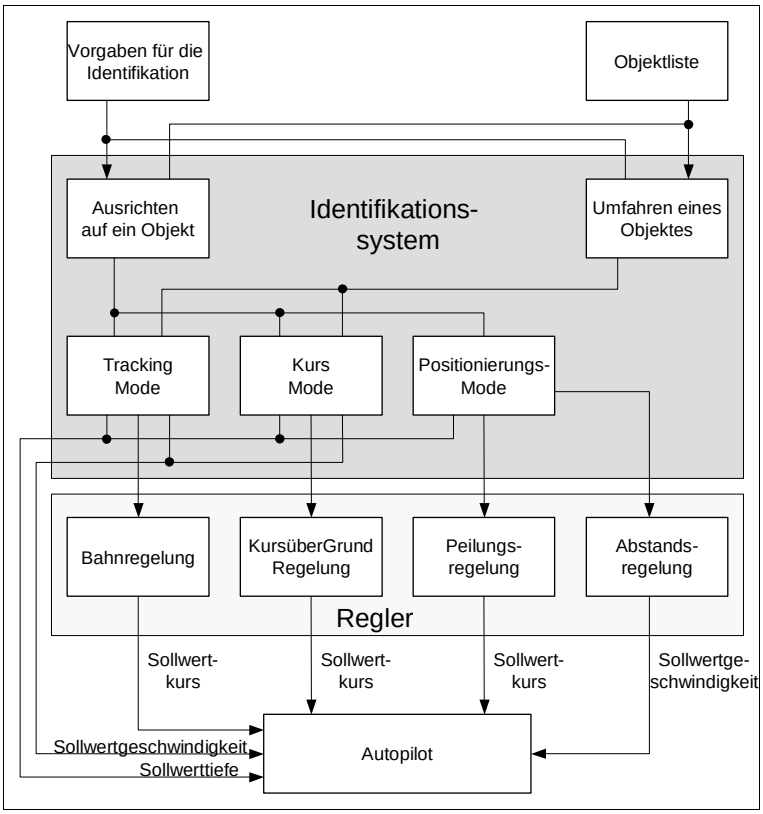


Abbildung 6.69: Struktur des Identifikationssystems

6.2.6.1.1 Ausrichten auf ein Objekt

Um eine Identifikation durchführen zu können, muss das Fahrzeug eine definierte Lage zum Objekt einnehmen und während der Identifikation halten. Dieser Modus wird als *Ausrichten auf ein Objekt* bezeichnet. Tritt in diesem Modus eine starke Seeströmung auf, muss das Fahrzeug in den Strömungsschatten des Objektes geführt werden, um eine exakte Position einnehmen und halten zu können.

Dabei werden unter Beachtung der Strömungsrichtung und eines Sicherheitsabstandes Gebiete um das Objekt definiert in welchen das Fahrzeug unterschiedliche Manövrieraufgaben erfüllen muss (siehe Abbildung 6.70). Befindet sich das Fahrzeug weit entfernt vom Objekt, soll es einen entsprechenden Zielpunkt anfahren (ZP Gebiete). Befindet sich das Fahrzeug näher am Objekt, wird es auf eine Sollbahn geführt (SB Gebiete). Wenn das Fahrzeug im Strömungsschatten liegt, erfolgt das Ausrichten auf das Objekt mit einem definierten Abstand (Ausrichten auf Objekt).

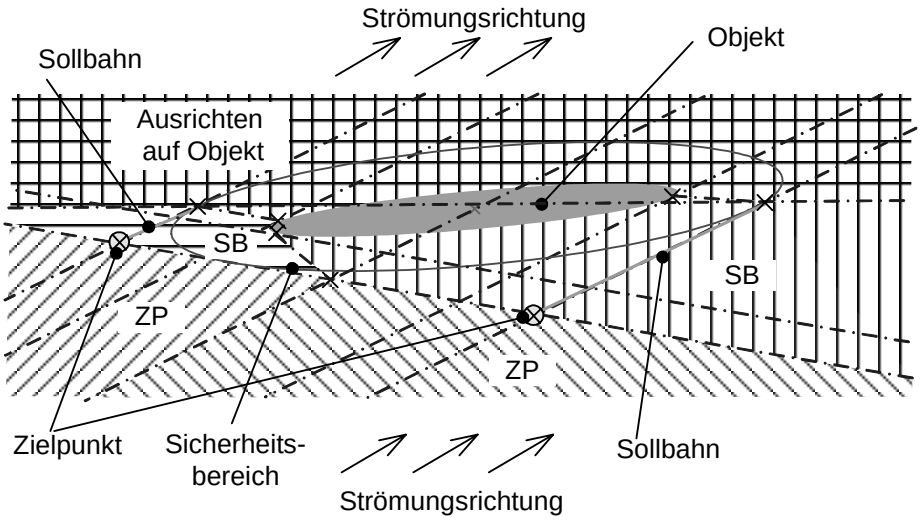


Abbildung 6.70 Ausrichten auf ein Objekt in Abhängigkeit der Strömung

6.2.6.1.2 Umfahren eines Objektes

Bei großräumigeren Objekten muss das Fahrzeug das zu identifizierte Objekt in einem definierten Abstand und in einer vorgegebenen Lage umfahren. Dieser Modus heißt *Umfahren eines Objektes.* Befindet sich das Fahrzeug weit entfernt von der vorgegebenen Solltrajektorie, wird es im Modus Sterring auf den kürzesten Weg an die Solltrajektorie geführt. Dann wird auf den Modus Tracking umgeschaltet und das Fahrzeug fährt im vorgegebenen Sollabstand um das Objekt herum (siehe Abbildung 6.71).

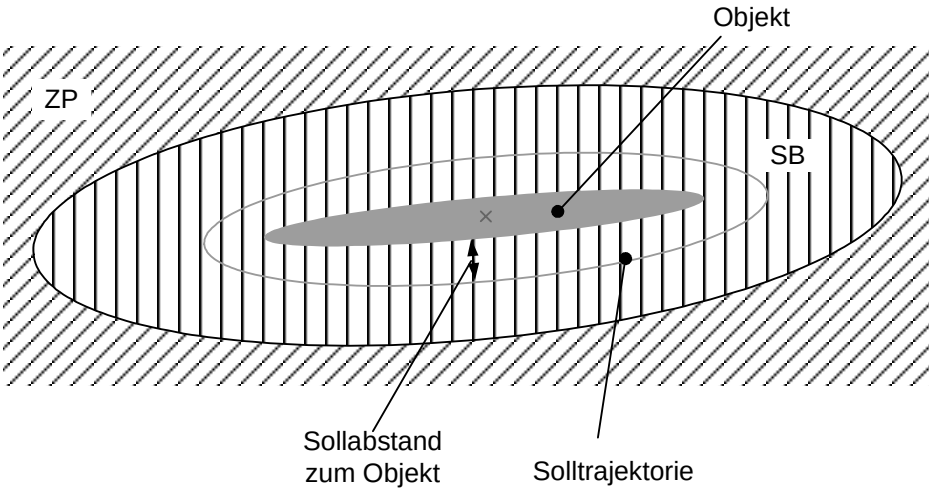


Abbildung 6.71: Umfahren eines Objektes

6.2.7 Zusammenfassung

In diesem Abschnitt wurde ein Konzept zur Führung des Unterwasserfahrzeuges in Sondersituationen vorgestellt. Die besonderen Merkmale des Fahrzeuges (vorhandene Rechenleistung und Sensorik, Fahrcharakteristik und Manövrierfähigkeit) bildeten die Hauptkriterien bei der Auswahl und Entwicklung der Algorithmen.

Durch den Einsatz von Graphenmethoden zur Ermittlung eines Routenvorschlages in der Wegeplanungsebene konnte die Forderung nach einer energieoptimalen Fahrweise leicht berücksichtigt werden. Die Einbeziehung der Seeströmung in die Kostenfunktion ermöglicht es, Gebiete mit starker Seeströmung von der Routenplanung auszuschließen.

Die Auswahl der untersuchten Methoden zur Generierung eines geometrischen Graphen erfolgte auf Basis der vorgegebenen Hindernisform (elliptischer Zylinder) und den rechentechnischen Restriktionen. Die durchgeführten Tests zeigten die Echtzeitfähigkeit der Graphenmethoden bei der Wegeplanung.

Das vorgestellte neue Verfahren zur geometrischen Konstruktion von Gradientenlinien für die Reaktive Ausweichsteuerung verbindet die Vorteile des Verfahrens der künstlichen harmonischen Dipol-Potentiale mit den Forderungen einer weg- und stellenergieoptimalen Fahrweise. Die Seeströmung sowie eine mögliche Geschwindigkeit der Hindernisse wurden bei der Konstruktion der Gradientenlinien durch Bildung einer Ersatzform für das Hindernis auf einfache Weise berücksichtigt.

Im Identifikationsmodus wurde das Operationsgebiet in Abhängigkeit der Objektgeometrie und der Seeströmung in separate Gebiete unterteilt in welchen spezielle Manöveraufgaben zu lösen sind. Hierzu werden spezielle Regler eingesetzt welche als Ausgang die Sollwerte für Kurs, Geschwindigkeit und den unterlagerten Autopiloten übergeben.

6.2.8 Literatur

[BH69] Bryson A. E. und Ho Y.-C.: *Applied Optimal Control*. USA: Ginn and Company, 1969.

[BHMM97] BRUTZMAN D; HEALEY T.; MARCO D.; MXGHEE B.: *The Phoenix Autonomous Unterwater Vehicle*. In Kortenkamp, D., Bonasso, P., & Murphy, R., editors, AI-Based Mobile Robots, chapter 13. MIT/AAAI Press. 1997.

[Boo04] http://www.boost.org.

[Bor04] http://www.borland.com.

[CGAL04] http://www.cgal.org/

[Dij59] DIJKSTRA E.W.: *A Note on Two Problems in Connexion with Graphs*, Numerische Mathematik, 1, pp.269-271, 1959.

[DJ00] DUDEK G. UND JENKIN M.: *Computational principles of mobile robotics*. New York, USA: Cambridge University Press, 2000.

[Ebe01] Eberly, D. H.: *3D Game Engine Design A practical Approach to Real-Time Computer Graphics*, San Diego, USA, Academic Press 2001.

[Fos02] FOSSEN T.I.: *Marine Control Systems. Guidance, Navigation and Control of Ships, Rigs and Underwater Vehicles*. Tapir Trykkeri, Trondheim, Norwegen, 2002

[GS98] GÄRTNER, B. UND SCHÖNHERR S. *Smallest Enclosing Ellipses An Exact and Generic Implementation in C++* Report B 98-05, Freie Universität Berlin, Institut für Informatik. http://www.inf.fu-berlin.de/inst/ubs/r-b-98-05. abstract.html 1998

[Gul95] GULDNER. J: *Intelligentes hierarchisches Regelungskonzept für autonome mobile Robotersysteme*. Düsseldorf, Germany: VDI-Verlag, 1995.

[Gul97] GULDNER J.: *Lokale Kollisionsvermeidung für mobile Roboter mittels künstlicher harmonischer Dipol-Potentiale*. at - Automatisierungstechnik, Volume 45, Issue 01, pp. 24-35 1997.

[Jac04] JACOBI M.: *Untersuchung von Bahnplanungsalgorithmen für das Autonome Unterwasserfahrzeug „DeepC“*, Studienjahresarbeit am Institut für

Automatisierungs- und Systemtechnik der TU Ilmenau. Ilmenau,2004.

[Kle97] KLEIN, R.: *Algorithmische Geometrie*. Bonn, Germany: Addison- Wesley-Longman, 1997.

[Mir98] MIRTICH, B., Efficient Algorithms for Two-Phase Collision Detection, in *Practical Motion Planning in Robotics*, K. Gupta and A. P. d. Pobil, Ed. Chichester: John Wiley & Sons Ltd, 1998, pp. 203-223.

[MSD04] http://msdn.microsoft.com/visualc.

[Nil71] Nilsson N.J: *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, 1971.

[Pap91] PAPAGEORGIOU M.: *Optimierung: statische, dynamische, stochastische Verfahren für die Anwendung*. München, Germany, R. Oldenbourg Verlag, 1991.

[Sch04] SCHERER S.: Untersuchungen von Algorithmen zur Fahrzeugführung bei Objektidentifikation des autonomen Unterwasserfahrzeuges „DeepC“, Studienjahresarbeit am Institut für Automatisierungs- und Systemtechnik der TU Ilmenau. Ilmenau,2004.

[SE03] SCHNEIDER P.J UND EBERLY D.H.*: Geometric Tools for Computer Graphics*. San Fancisco, USA: Morgan Kaufmann, 2003.

[SLL02] SIEK J.G., LEE L.-Q., UND LUMSDAINE A.: The Boost Graph Library: User Guide and Reference Manual. New York, USA: Addison Wesley, 2002.

[Sps04] http://www.sbs.com.

[Ste95] STENTZ A.: *The focussed D* Algorithm for Real-Time Replanning*. - In: Proceedings of the International Joint Conference on Artificial Intelligence, Montreal, Canada, August 1995.

[Wei03] WEISSTEIN E.W.: *CRC Concise Encyclopedia of Mathematics:* Boca Raton, USA: Chapman & Hall/CRC, 2003.

[WHI99] WATANABE K., HASHEM M.M.A. UND IZUMI K.: *Global Path Planning of Mobile Robots as an Evolutionary Control Problem*, European Control Conference, Karlsruhe, Germany, 1999.

6.3 Maschinelles Lernen

6.3.1 Aufgaben des maschinellen Lernmodul

Das Ziel dieses Arbeitspaketes bestand in der Untersuchung von Möglichkeiten einer dreidimensionalen Steuerung eines Unterwasserfahrzeuges (AUV) durch Hindernisse bei der Fahrt von einem Startpunkt zu einem stationären Zielpunkt. Das AUV verfügt nur über eine begrenzte Anzahl von Sensoren, einschließlich Sonar und kann seine Geschwindigkeit und Fahrtrichtung in jedem Entscheidungsschritt aktualisieren. Das Unterwasserfahrzeug soll zunächst in der virtuellen Welt, auf der Grundlage der Handlungen eines Operateurs, Strategien zur Vermeidung von Kollisionen erlernen, wenn es ein Hindernis entdeckt hat (d.h. Reiz-Reaktion (stimulus-response) Regeln) [KO04]. Das System erlernt somit keinen spezifischen Weg, sondern Regeln, die reaktiv über eine Bewegungsrichtung in jedem Zeitschritt entscheiden, so dass sie zukünftig genutzt werden können, um dem Manövermanagement in ähnlichen Situationen in der realen Welt eine autonome Führung des Fahrzeuges zu ermöglichen.

6.3.2 Einführung

Das Manövrieren durch verschiedenartigste Unterwasserhindernisse wie z.B. Felsen, Wracks, Unterwasserbauten oder Minen ist eins von vielen wichtigen Fähigkeiten des Verhaltens eines autonomen Unterwasserfahrzeuges. Eine Methode zur Erzielung einer robusten Unterwasserfahrzeugführung besteht in einer prädiktiven Bahnplanung [GG90]. Die Echtzeitfähigkeit solcher Systeme genügt jedoch häufig nicht den hohen Anforderungen, die bei der Steuerung eines autonomen Unterwasserfahrzeuges benötigt werden. Reagierende Systeme, in denen Reiz-Reaktion Regeln das Verhalten des Unterwasserfahrzeuges bestimmen, können diese Echtzeitfähigkeit besser realisieren und in einer Vielzahl von Situationen einfach und schnell umgesetzt werden.

Ein interessantes und schwieriges Problem besteht in der Entwicklung von Methoden, die es gestatten, effektive Regeln für solche reagierenden Systeme zu ermitteln. Im Projekt DeepC wurde dazu das Programmsystem FuzzyMod® eingesetzt [DKO97], [OW02], ein Maschinelles Lernsystem, das auf dem ID3-Algorithmus basiert und Klassifikationslernprobleme lösen kann. Es wird verwendet, um leistungsstarke reagierende Strategien zur Navigation und Kollisionsvermeidung zu erlernen. Diese Aufgabenstellung erfordert, ein AUV durch ein nach dem Zufallsprinzip erzeugtes, dichtes Hindernisfeld zu steuern, um ohne Kollision von einem vorgegebenen Startpunkt zu einem Zielpunkt zu gelangen. Das AUV hat eine begrenzte Anzahl von Sensoren, einschließlich Sonar und soll seine Geschwindigkeit und seine Fahrtrichtung in jedem Entscheidungsschritt den Gegebenheiten anpassen. Die Strategie oder der Plan, besteht aus einem Satz gelernter reagierender Regeln (d.h. Reiz-Reaktion Regeln), die eine Zuordnung von Sensormesswerten zu Steuerkommandos vornehmen.

Die erzielten Ergebnisse zeigen, dass das Unterwasserfahrzeug mit diesen Regeln in der Lage ist, in einer unbekannten Umgebung mit zufällig generierten Hindernissen, einwandfrei von einem beliebigen Startpunkt zu einem beliebigen Zielpunkt zu navigieren und Hindernisse zu vermeiden.

6.3.3 Vorbereitungsarbeiten

6.3.3.1 Anforderungen an das Lernmoduls

Die erste Aufgabe war die Ermittlung der Anforderungen, die durch das Manövermanagement an das Systems gestellt werden, um die Schnittstelle mit den anderen Modulen des Unterwasserfahrzeuges festlegen zu können. Das Lernmodul ist dem

Ausweichmodul zugeordnet. Es dient der Bereitstellung von Regeln für das Ausweichmodul und wird nicht direkt im Fahrzeug integriert (siehe Abbildung 6.72).

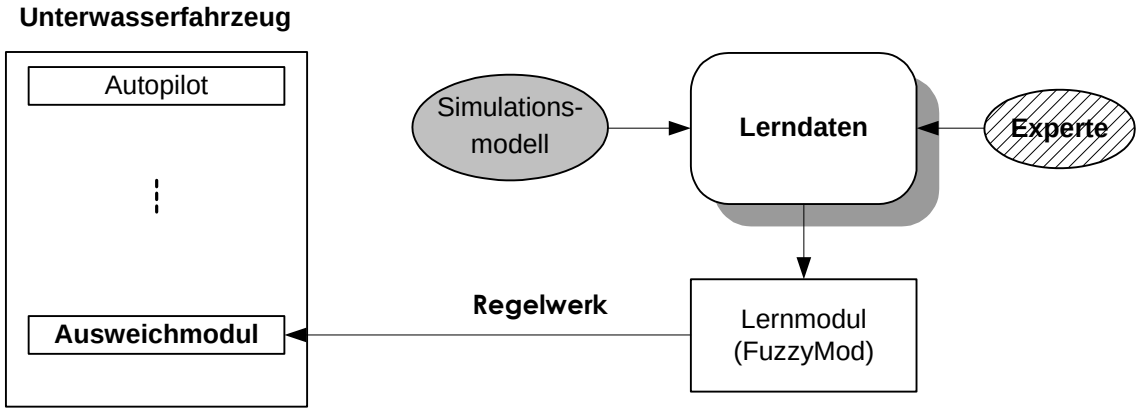


Abbildung 6.72: Systemschnittstelle (Lernmodul - Unterwasserfahrzeug)

Zunächst wurde die Software Requirements Specification (SRS) für das Lernmodul des Unterwasserfahrzeuges erstellt, indem alle für das Lernen und die Manöverdurchführung notwendigen Parametern erfasst und zusammengestellt worden sind. Die Eingabeparameter des Lernmoduls beinhalten die

- Objektdaten (Position, Geschwindigkeit und die Bewegungsrichtung von Hindernissen),
- Zustandsdaten des Fahrzeuges (Position, Lage, Geschwindigkeiten),
- Umweltinformationen,
- Sollbahninformation und alle
- Daten, die zur Ausführung von Fahrmanövern intern generiert werden (Manöverkommandos).

Da das Manöververhalten des Fahrzeuges auch nach Energieverbrauch und Zeitverhalten optimiert werden soll, gehört die noch zur Verfügung stehende Energie ebenfalls zu den Eingabeparametern des Lernmoduls.
Zusätzlich sind für eine optimale Manöverdurchführung als weitere notwendige Eingabeparameter die Zustände der Motoren und Sensoren in Betracht zu ziehen.
Diese Informationen benötigt das Lernmodul, um Manöverstrategien auch bei Teilausfall der Motoren und Sensoren entsprechend lernen und ausführen zu können.

Die Ausgabeparameter des Lernmoduls beinhalten

- den berechneten Kurs und
- die Geschwindigkeit des Unterwasserfahrzeuges.

Alle für das Lernmodul benötigten Informationen werden durch das Manövermodul bereitgestellt. Das Regelwerk des Lernmoduls liefert die Sollwerte, die vom Autopiloten benötigt werden, steuert aber nicht direkt die einzelnen Komponenten des Fahrzeuges, wie Motoren, Ruder usw.

6.3.3.2 Auswahl von geeigneten Lernverfahren

Es wurden unterschiedliche Methoden des Maschinellen Lernens, die zum Lernen von Manöverstrategien des Fahrzeuges geeignet sind, ausgewählt und analysiert. Im Ergebnis der Untersuchungen haben sich zwei wesentliche Möglichkeiten herauskristallisiert. Die erste beschreibt das zu lernende Manöververhalten in Form von Regeln, wobei als Lernverfahren der ID3- bzw. C4.5-Algorithmus von Quinlan verwendet wird [Qui92], der auf der Grundlage

von informationstheoretischen Zusammenhängen in den verfügbaren Daten einen minimal diskriminierenden Entscheidungsbaum und daraus Regeln erstellt. Die Regeln werden zur Verbesserung der Abbildungsgenauigkeit in einem Fuzzy-System verarbeitet, wobei stückweise lineare Zugehörigkeitsfunktionen gebildet und diese nach Erstellung der Regeln optimiert werden [OW02].
Das zweite geht von der Verwendung von Künstlichen Neuronalen Netzen aus und versucht den Zusammenhang zwischen der aktuellen Situation des AUV und den daraus resultierenden Steuerhandlungen durch ein Multilayer Perceptron (MLP) abzubilden [KO02]. Nach dem Training des neuronalen Netzwerkes wird mit bekannten Verfahren (wie Brainne) eine Regelextraktion vorgenommen. Als Grundlage des Lernvorganges (supervised learning) sollen für beide Methoden zunächst Steuerhandlungen dienen, die durch einen Experten bei der Steuerung des AUV in der virtuellen Welt erzeugt wurden.

6.3.3.3 Auswahl zu lernender Manöverstrategien

Bei der Auswahl, der zu untersuchenden Manöverstrategien sind alle vorhersehbaren Situationen, in denen ein Ausweichmanöver notwendig wird, berücksichtigt worden. In einer dreidimensionalen Umgebung des Fahrzeuges sind folgende Ausweichsituationen vorstellbar:

Das AUV muss geradlinig (auf schnellstem Weg) von einem Startpunkt zu einem Zielpunkt fahren können.

- Sollte eine Mission die Inspektion eines bestimmten Objektes erfordern, muss das AUV in der Lage sein, dieses Objekt zu umrunden.
- Das AUV muss in der Lage sein, einen vorgegebenen Abstand zu einem bestimmten Objekt einzuhalten und dabei eine bestimmte Lage relative zum Objekt einzunehmen, z.B. beim Manövrieren in einer Schlucht, wobei das Fahrzeug bestimmte Abstände in vier Richtungen (rechts, links, oben und unten) einhalten muss oder bei der Verfolgung von festen (Pipelines, Kabel o.a.) bzw. beweglichen Objekten (Schiffe u.a.).

6.3.4 Vorgehensweise der Lernstrategien

Wegen der Kompliziertheit der Wissensermittlung in Verbindung mit dem Entwurf von Expertensystemen, wird nach Maschinellen Lernverfahren geforscht, die den Wissensermittlungsprozess automatisieren und die Nutzung der zugänglichen Informationsquellen erweitern [Vac94].

Die Wahl einer passenden Lernmethode hängt von der Natur der zu lösenden Aufgabenstellung und von der Art des vorhandenen Wissens ab [RS90]. Wenn die zu lösende Aufgabe Klassifikationscharakter hat und viele Trainingsbeispiele vorhanden sind, dann können induktive Lernmethoden verwendet werden, um Klassifikationsregeln zu erlernen.
Soll bereits vorhandenes theoretisches Wissen durch wenige neue Beispiele modifiziert bzw. verbessert werden, können Explanation-Based-Learning (EBL) Methoden angewendet werden.

Viele interessante praktische Probleme, die dem automatisierten Lernen zugänglich sein könnten, passen jedoch nicht immer in diese Kategorien. Dazu gehört z.B. die Lösung sequentieller Entscheidungsaufgaben, bei denen weder eine Datenbank von Beispielen, noch genügendes theoretisches Wissen vorliegt, auf das sich traditionelle Maschinelle Lernmethoden stützen könnten. Ein Ausweg zur automatischen Regelgewinnung besteht dann darin, manuell Lerndaten mit Hilfe eines Simulations-Modells zu erzeugen und induktive Lernmethoden anzuwenden [GG90].

Die Vorgehensweise, die hier beschrieben wird, reflektiert eine bestimmte Methode für das Lernen mit einem Simulationsmodell. Die Notwendigkeit für die Anwendung dieser Methode besteht darin, dass Fehler bei Versuchen mit realen Systemen teuer werden können, gefährlich sind oder wie im vorliegenden Fall das reale System noch nicht existiert. Da das Lernen Experimente mit taktischen Plänen erfordert, die gelegentlich nicht akzeptable Resultate produzieren können, wenn sie auf die reale Welt angewendet werden, wird angenommen, dass hypothetische Pläne in einem Simulationsmodell ausgewertet werden(Abbildung 6.73).

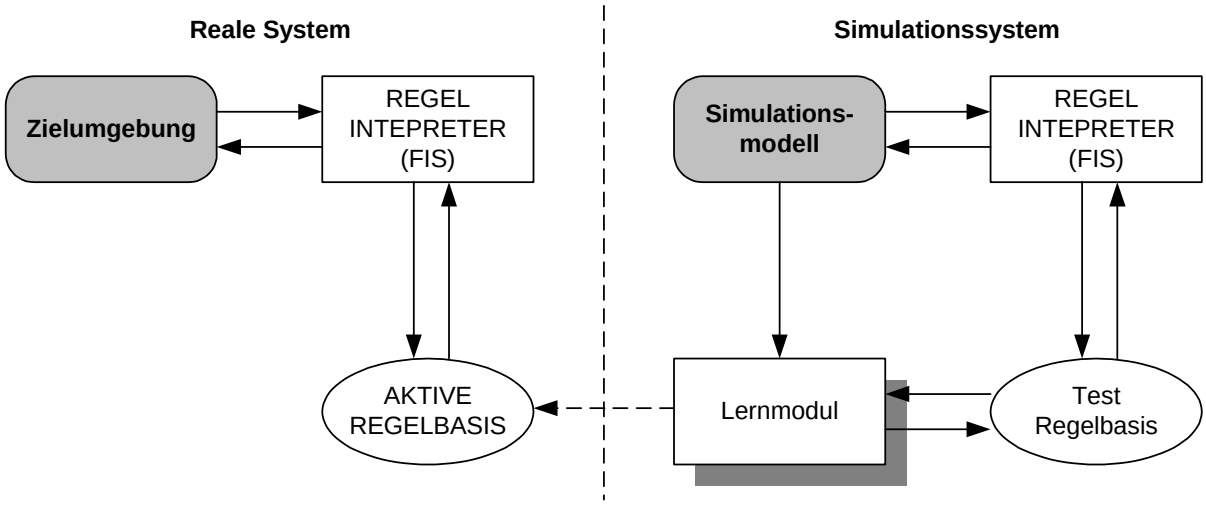


Abbildung 6.73: Lernen mit einem Simulationsmodell

Vorhergehende Studien mit Kleinrobotern haben gezeigt, dass das durch die Simulation erlernte Verhalten robust ist und in der realen Umgebung angewendet werden kann, wenn die Simulation unter verschärfteren Bedingungen durchgeführt wird, als sie in der realen Welt auftreten (z.B. Störungen der Sensorsignale)[RS90].

6.3.5 Beschreibung des Lernmoduls

Die Hauptzielsetzung des experimentellen Systems besteht darin, Lerndaten zu erzeugen, während das Unterwasserfahrzeug manuell durch Hindernisse in einer virtuellen Welt navigiert wird. Lerndaten (Sensorwerte und Steuerhandlungen, die durch einen Experten bei der Steuerung des Unterwasserfahrzeugs erzeugt werden) werden für ausgewählte Szenarien der Hindernisvermeidung in der virtuellen Umgebung mit dem Modell, das durch die TU Ilmenau erstellt wurde und mit dem das dynamische Verhalten des Fahrzeuges unter Wasser nachgebildet wird, erfasst. Diese Daten dienen als Grundlage für den Lernvorgang. Nach dem Lernen kann das Unterwasserfahrzeug mit Hilfe der gelernten Regelbasis und eines Fuzzy-Regelverarbeitungssystems selbständig durch Hindernisse in einer unbekannten Umgebung manövrieren.

Der Versuchsaufbau für das Lernen und Testen des Reglers ist in Abbildung 6.74 dargestellt. Das Modell kann in zwei Modi geführt werden, einem für die Datenerfassung (Schalter F1 ist auf C) und einem anderen zum Testen des entworfenen Reglers (Schalter F1 auf T). Wie in der Abbildung zu sehen, besteht das Simulationsmodell aus vier Komponenten:

- dem Datenaufbereitungsmodul,
- dem Regler, der auf dem Fuzzy Inference System (FIS) basiert,
- dem Experten (im Datenerfassungsmodus),
- dem Unterwasserfahrzeug und seiner Umgebung.

Alle diese Komponenten, die Datenerfassungs- und Lernprozesse werden in den folgenden Abschnitten, beginnend mit dem Datenaufbereitungsmodul, detailliert beschrieben. Besonders der Datenerfassungsprozess muss geschickt durchgeführt werden, da der Experte gezwungen ist, bei der Ausführung von Ausweichmanövern alle anderen Informationen, wie Ansicht von oben, Gesamtansicht, usw. zu ignorieren und nur die Sonarinformation zu benutzen. Dies erfordert sehr viel Übung und Geschick.

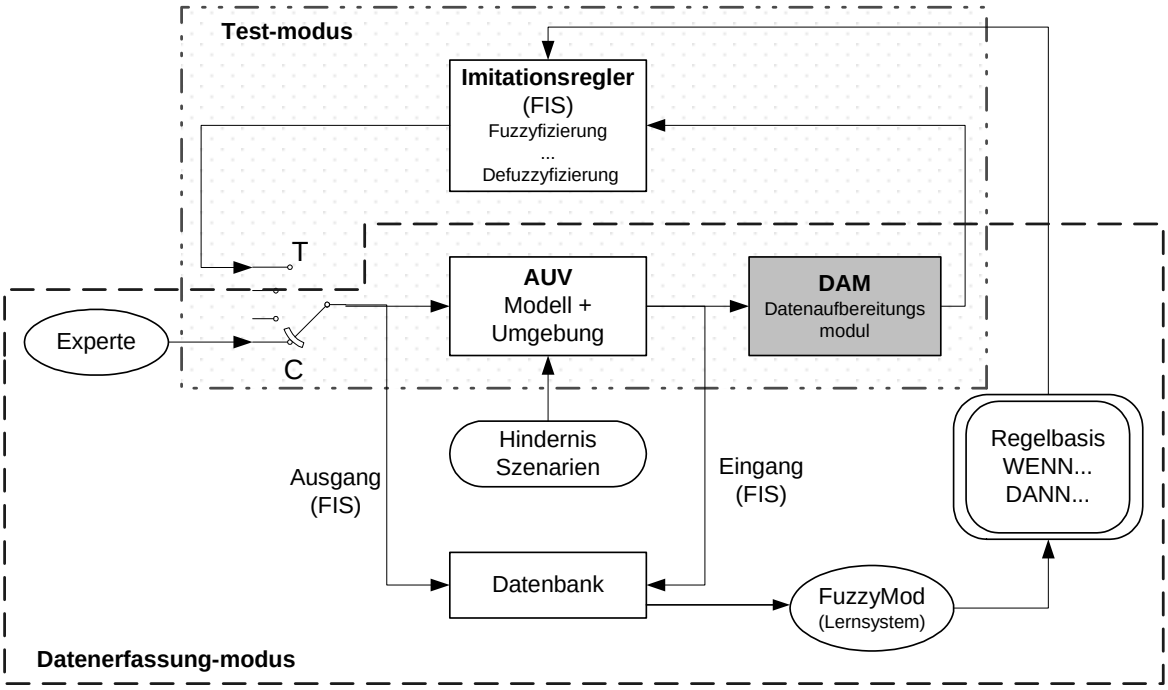


Abbildung 6.74: Versuchsaufbau für Lern- und Testzwecke

6.3.5.1 Datenaufbereitungsmodul

Dieses Modul dient der Umwandlung von absoluten Fahrzeugdaten wie Absolutposition, -geschwindigkeit, -sonardaten in relative Daten wie dem Winkel und dem Abstand zum Ziel oder dem Winkel und dem Abstand zum Hindernis (Abbildung 6.75). Dies reduziert drastisch die zur Datenerfassung durchzuführenden Experimente. Dies ist nachvollziehbar, da aus absoluten Variablen eine geringere Anzahl relativer Variabler erzeugt wird. So entstehen z.B. aus den sechs absoluten Variablen (AUV-Position(x,y,z) und Hindernisposition(x_h,y_h,z_h)) nur zwei relative variable Winkel(α) und Abstand(r) zum Hindernis. Es wird angenommen, dass das Unterwasserfahrzeug seine Position mit einer bestimmten Genauigkeit kennt, und dass die feste Zielposition auch bekannt ist (die durch globale Planung erzeugten temporären Zielpunkte werden hier auch als fest bezeichnet, weil sie im Voraus bestimmt werden und a priori bekannt sind). Das Unterwasserfahrzeug hat interne, oder virtuelle Sensoren, um seinen eigenen Zustand zu bestimmen. Um das Unterwasserfahrzeug optimal steuern zu können, werden die folgenden Sensoren benötigt (das sind die Variablen, die das Datenaufbereitungsmodul zumindest zur Verfügung stellen soll):

- Abstand zum Ziel: Der Abstand zum Zielpunkt (es kann auch ein temporärer Zielpunkt sein). Nimmt Werte von 0 bis 4000 m an.
- Geschwindigkeit: Die momentane Geschwindigkeit des Unterwasserfahrzeuges. Nimmt Werte von –1.5 bis 3 Knoten an.
- Winkel zum Ziel: Die Richtung zum Zielpunkt relativ zum Unterwasserfahrzeug. Nimmt Werte von –180 bis 180 Grad an. 0 Grad bedeutet direkt in Richtung des Unterwasserfahrzeuges, und 180 Grad bedeutet entgegengesetzt der Richtung des Unterwasserfahrzeuges.

- Letzte Richtungsänderung: Die momentane Richtungsänderung des Unterwasserfahrzeuges. Nimmt Werte von -30 bis 30 Grad an.
- Aktive Sonarzelle_N: eine der 24 aktiven Sonarzellen für die Hindernisvermeidung. Jede Zelle nimmt Werte von 0 bis 200 an und repräsentiert den Abstand des Objekts im Sichtbereich. Wenn kein Objekt in dem Bereich entdeckt wurde, wird ein spezieller Wert 220 (indiziert freie Sicht) ausgegeben.

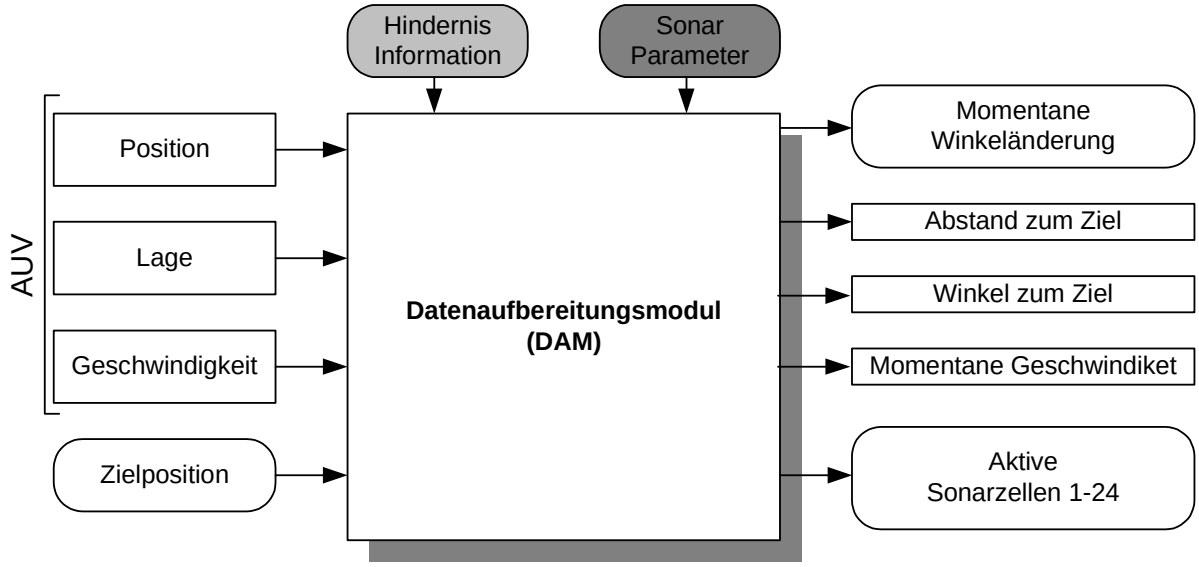


Abbildung 6.75: Datenaufbereitungsmodul

Das Datenaufbereitungsmodul verarbeitet die Daten in zwei Schritten. Zuerst werden alle Abstände und Richtungen der Oberflächenpunkte des Hindernisses (a_{new}) wie in Abbildung 6.76 dargestellt, von den Eingängen, der Hindernisposition (x_h,y_h,z_h), der Unterwasserfahrzeug-Position (a_{auv} = x,y,z), dem Teilungswinkel (angx), dem Hindernisradius (r_h) und der Hindernishöhe (h_h) berechnet.

Dies geschieht durch folgende Prozedur:

Es werden systematisch Punkte auf der Hindernisoberfläche mit Hilfe von angx = θ ausgewählt und anschließend für jeden selektierten Oberflächenpunkt der Winkel und der Abstand relativ zum Unterwasserfahrzeug (α_z, α_y) berechnet.

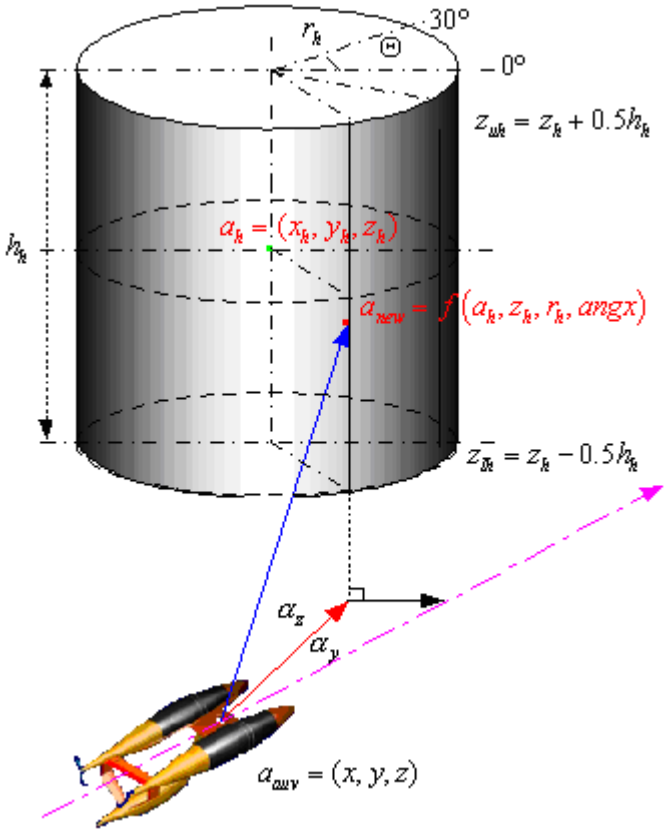


Abbildung 6.76: Abstände und Richtungen der Oberflächenpunkte

Im nächsten Verarbeitungsschritt wird der Sonarsichtbereich (Abbildung 6.77) in die Zellen 1-24 aufgeteilt und für jeden selektierten Oberflächenpunkt des Hindernisses der Winkel und der Abstand relativ zum AUV (α_z, α_y) berechnet. Alle Oberflächenpunkte, die in der Richtung des Sonarzelle(i) liegen, werden zusammengefasst und der Sonarzelle wird der minimale Abstand von den selektierten Oberflächenpunkten zugewiesen. Wenn kein Objekt in dem Bereich entdeckt wurde, wird ein spezieller Wert 220 (indiziert freie Sicht) ausgegeben.

Zusammengefasst, umfasst der Algorithmus folgende Schritte:

(a) Teilen des Sonarbereiches in Zellen (1-24) (Abbildung 6.77)

(b) Repräsentative Punkte auf der Oberfläche des Hindernisses bestimmen (Abbildung 6.76)

(c) Für jeden Punkt den Abstand und die Richtung bezüglich des AUV’s berechnen

(d) Für alle Punkte, die in einer Zelle liegen, den minimalen Abstand bestimmen und diese Zelle mit diesem Abstandswert bewerten.

(e) Wenn kein Hindernis in der Richtung einer Sonarzelle liegt, dann wird die Sonarzelle mit 220 bewertet.

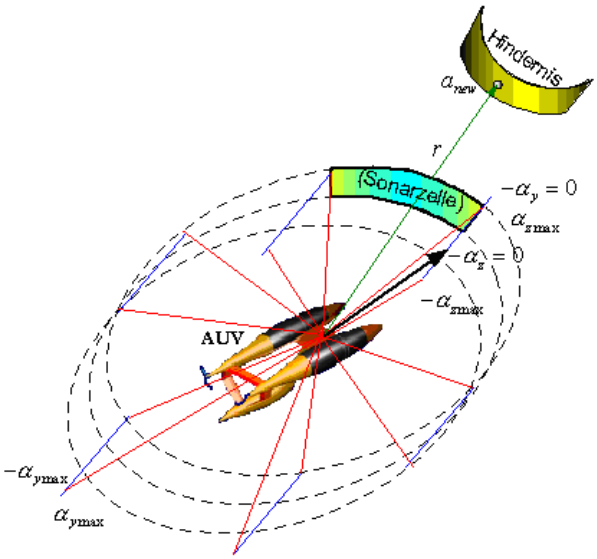


Abbildung 6.77: Zerlegung des Sonarsichtbereiches in Zellen

$\alpha_{z\max}$, $\alpha_{y\max}$ und r sind der maximale Winkel in z-Richtung, der maximale Winkel in y-Richtung bzw. der Abstand zum Oberflächenpunkt a_{new}. Die Nummerierung der Sonarzellen erfolgt wie in Abbildung 6.78 dargestellt. In dieser Anwendung wurde der Sichtbereich des Sonars auf $-60<a_y<60$ bzw. $-10<a_z<10$ begrenzt.

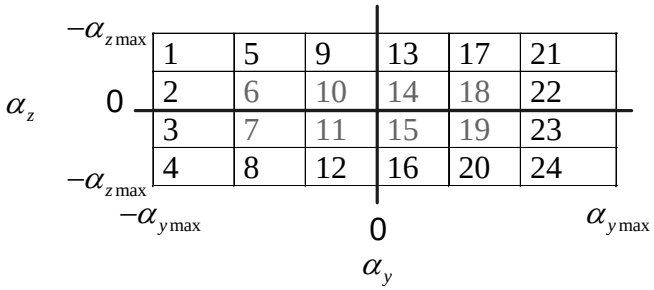
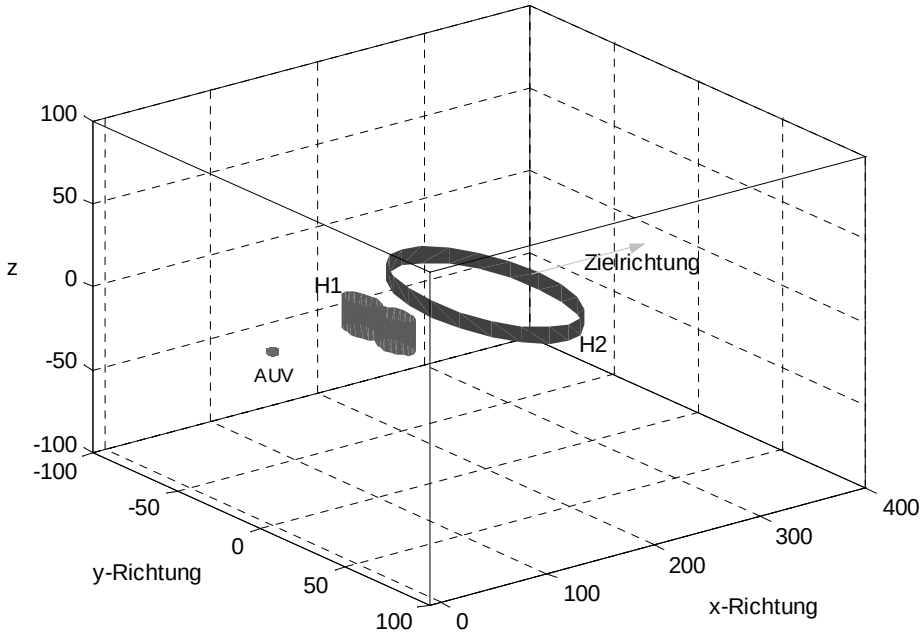
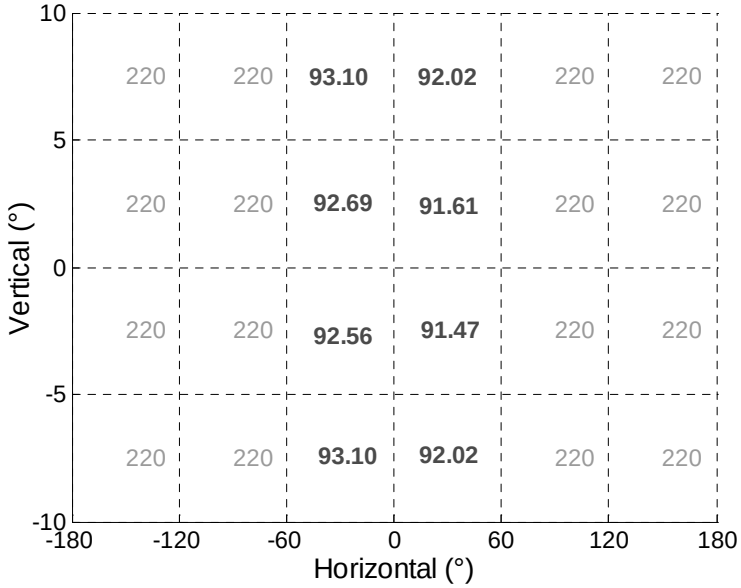


Abbildung 6.78: Nummerierung der Sonarzellen

Die Ergebnisse des Aufbereitungsmoduls zeigen beispielsweise Abbildung 6.79. In Abbildung 6.79 befindet sich das Hindernis direkt vor dem Unterwasserfahrzeug und in Abbildung 6.80 befindet sich das Hindernis links vor dem Unterwasserfahrzeug. Die Diagramme in Abbildung 6.79 (b) und Abbildung 6.80(b) zeigen die minimalen Abstände in den entsprechenden Sonarzellen. Die Ergebnisse sind wie sie sein sollen.

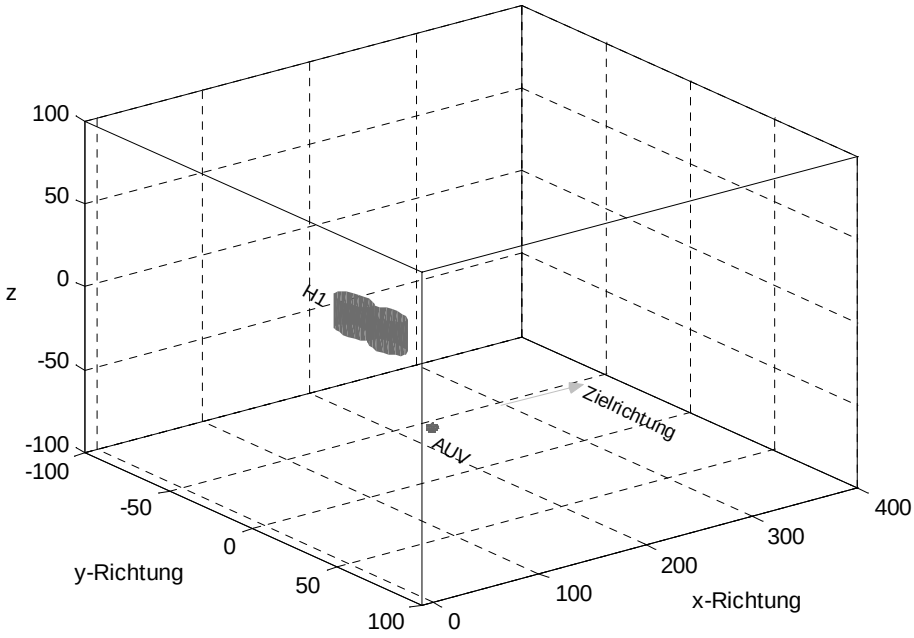


(a) Hindernisposition (H1, H2) relativ zum AUV

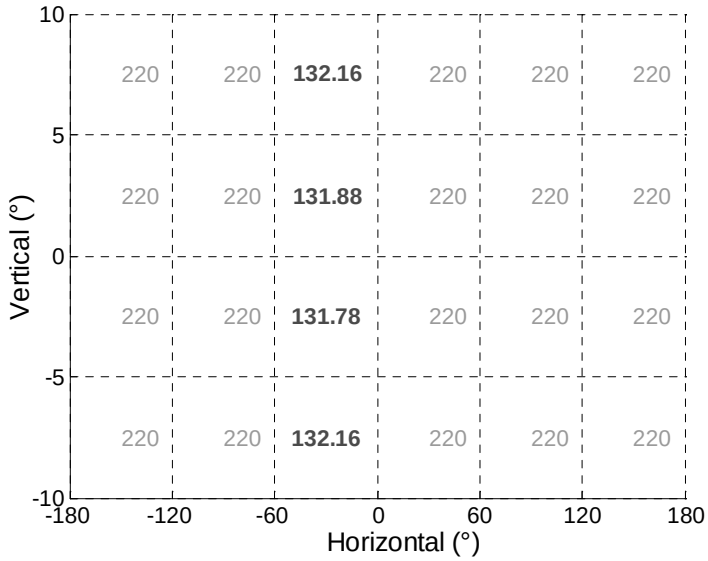


(b) Sonarsicht.

Abbildung 6.79: Hindernis direkt Vorn



(a) Hindernisposition relativ zum AUV



(b) Sonarsicht.

Abbildung 6.80: Hindernis Vorn-Links

6.3.5.2 Fuzzy-Regelbasierter Regler

Um das Verhalten eines Experten nachzubilden, ist ein Imitationsregler erforderlich. Der Imitationsregler, der hier eingesetzt wird, basiert auf einem Fuzzy-Inferenzsystem, d.h. er führt die Funktionen Fuzzyfizierung, Inferenz und Defuzzyfizierung durch. Der Regler besteht aus den in der
Abbildung 6.81 dargestellten Eingangs- und Ausgangsgrößen. Er bekommt Daten vom Datenaufbereitungsmodul.

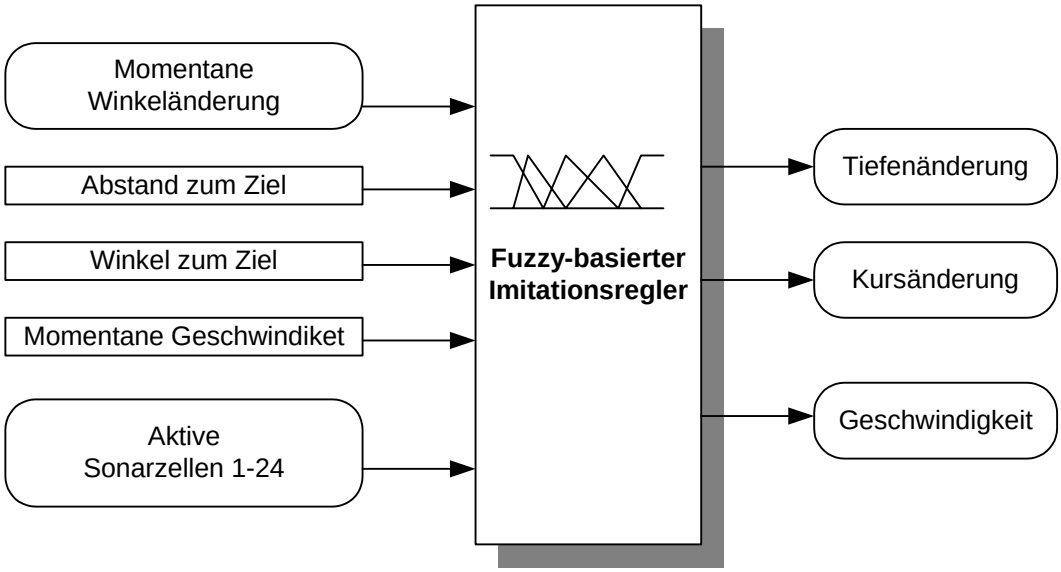


Abbildung 6.81: Fuzzy-basierter Imitationsregler

Zusätzlich zu den Funktionen wie Fuzzyfizierung, Inferenz und Defuzzyfizierung, erfordert ein Fuzzy-basierter Regler eine Regelbasis. Die Regelbasis wird automatische aus Lerndaten mit dem Tool FuzzyMod® generiert, wie es in den folgenden Abschnitten beschrieben wird. Um FuzzyMod® anwenden zu können, müssen vorher Daten über das System erfasst werden. Der nächste Abschnitt behandelt das Verfahren zur Datenerfassung in der virtuellen Welt.

6.3.5.3 Datenerfassung

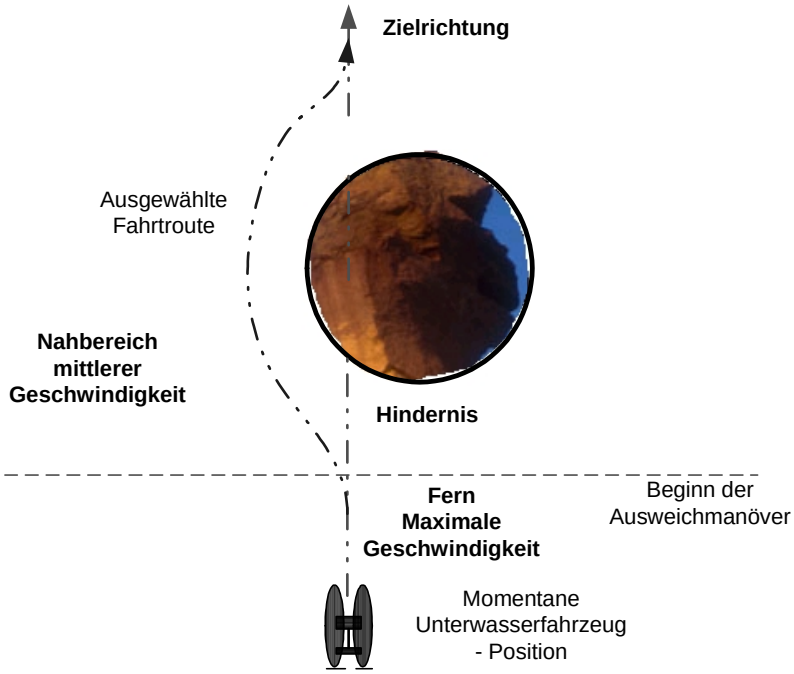
Zur Datenerfassung sind mehrere Experimente erforderlich, um alle möglichen Ausweichsituationen simulieren zu können. In jedem Experiment werden die Werte der Sensoren abgelesen, während sich das Unterwasserfahrzeug in der Simulationsumgebung befindet. Das Unterwasserfahrzeug wird während jedes Experimentes gefahren. Von den Sensoren abgelesene Daten werden während eines Experimentes in einer Datenbank gespeichert. Nachdem das Experiment abgeschlossen ist, werden mit der dazu entwickelten Software die Daten in eine Form umgewandelt, die für das Training oder die Validierung des Imitationsreglers benutzt werden kann. Für das Lernsystem FuzzyMod® erhalten die Daten z.B. folgende Form:

Eingang 1	Eingang 2	---	Eingang N	Output
⋮	⋮		⋮	⋮

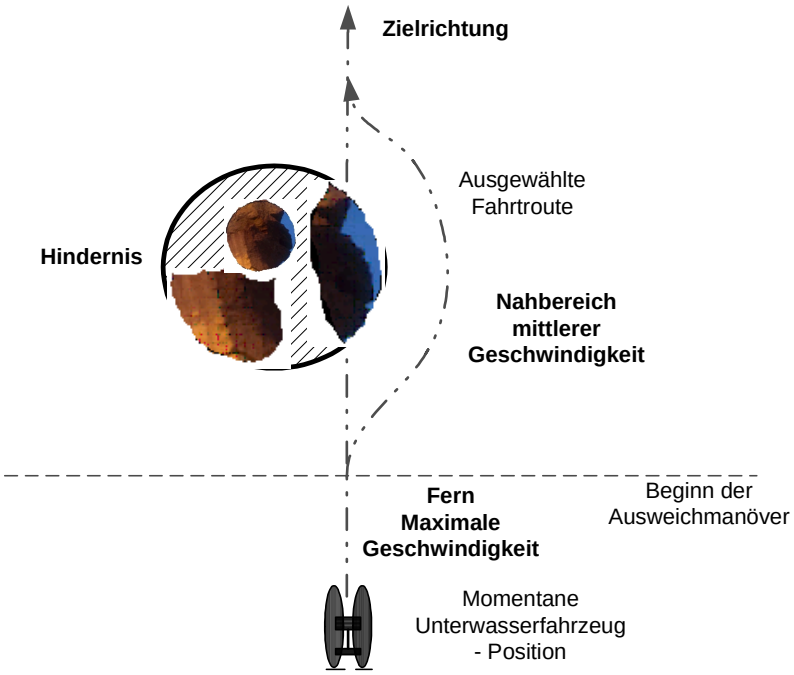
Abbildung 6.82: Datenformat für das Lernsystem FuzzyMod®

Das Hauptziel jedes Experiments ist es, die von den Sensoren gelieferten Daten zu erfassen, die für einen spezifischen Zustand des Unterwasserfahrzeuges in seiner Umgebung repräsentativ sind. Der Imitationsregler muss Regeln erhalten, die die primären Steuerfunktionen wie Rechts -, Links -, nach oben bzw. nach unten abbiegen enthalten, wenn ein Hindernis entdeckt wurde. Hier sind die für jedes Experiment zu verändernden Parameter und eine kurze Beschreibung wie er in dem Experiment implementiert werden soll angegeben:

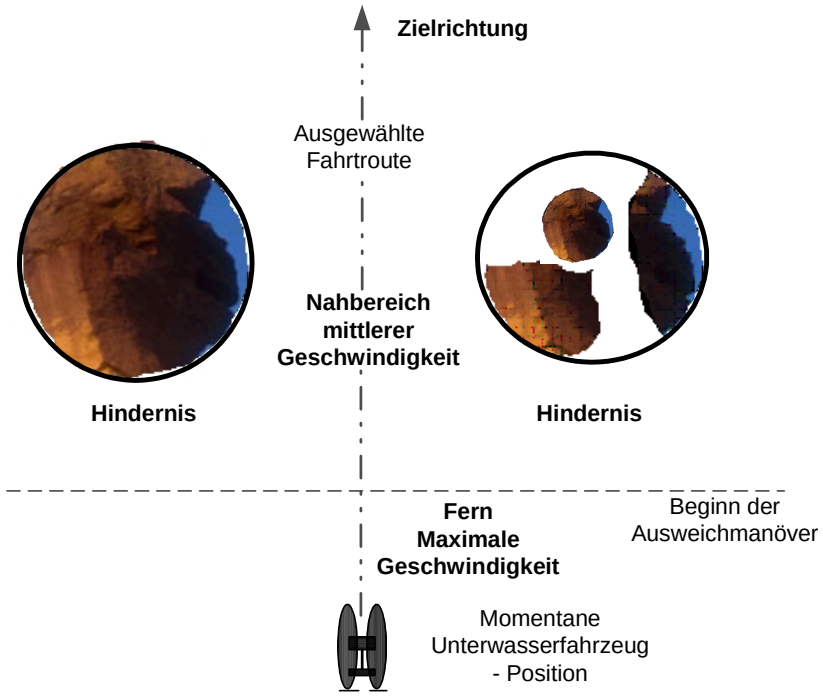
1) **Hindernisposition:** Es gibt sechs Zustände, wie ein Hindernis relativ zum Unterwasserfahrzeug liegen (oder nicht) kann: Links, Rechts, Zentrum, Oben, Unten, Korridor oder Keins (Abbildung 6.83 a-d). Links bedeutet, die linken Sonarzellen haben ein Objekt registriert. Rechts bedeutet, die rechten Sonarzellen haben ein Objekt registriert. Zentrum bedeutet, beide linke bzw. rechte Sonarzellen haben ein Objekt registriert. Keins heißt, alle Sonarzellen haben kein Objekt registriert.



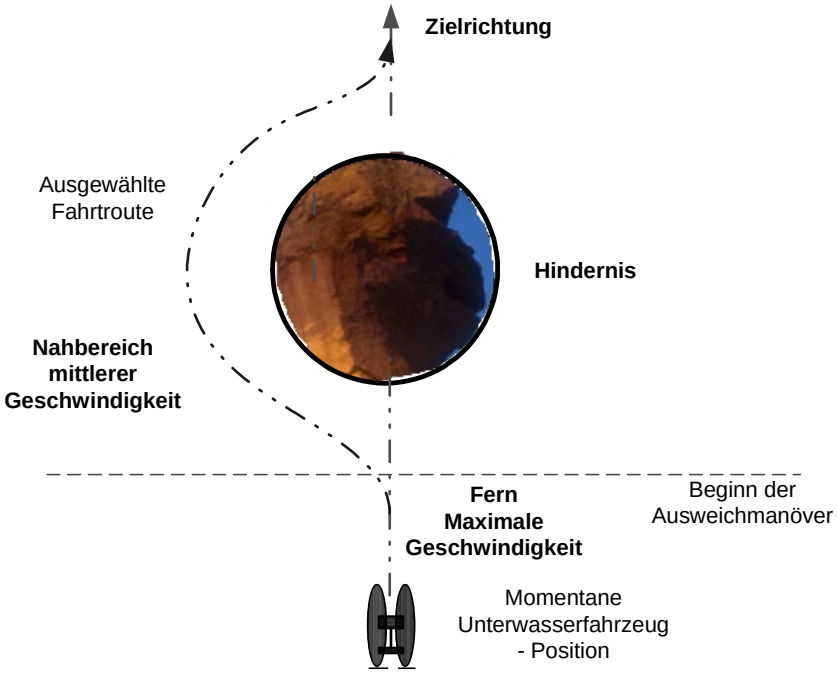
(a) Hindernis Rechts



(b) Hindernis Links



(c) Korridor



(d) Hindernis im Zentrum-Vorn

Abbildung 6.83: Mögliche Hindernisposition relative zum Unterwasserfahrzeug

2) **Abstand zum Hindernis:** Da das Unterwasserfahrzeug während jedes Experiments in Bewegung sein wird, werden sich die Abstände zu den Hindernissen ständig ändern. Ein Sicherheitsabstand wird beibehalten.

3) **Zielposition:** Die Zielposition wird relativ zum Unterwasserfahrzeug von 0 bis 360 Grad verändert und jedes Mal wird das Unterwasserfahrzeug gedreht und in die richtige Richtung navigiert.

In jedem Experiment wird das Unterwasserfahrzeug von einem Experten mit Hilfe eines Joysticks durch die Hindernisse gefahren. Es wird empfohlen, dass der Experte nur den Ausschnitt aus der virtuellen Welt sieht, den auch das Sonar erfasst, d.h. der Experte kann nur sehen was das Unterwasserfahrzeug sehen kann, so dass nur die Sensoren, die der Imitationsregler zur Verfügung hat, dem Experten zur Verfügung stehen. Nach Abschluss der Datenerfassung werden die Daten zum offline-Lernen der Regeln für den Imitationsregler mit FuzzyMod® benutzt. Da durch die Steuerregeln eine Zuordnung zwischen der Situation des AUV und den Steuerhandlungen des Operateurs erfolgen soll, kann der Algorithmus für die Ermittlung statischer Zusammenhänge in FuzzyMod® verwendet werden.

6.3.5.4 Regelgenerierung mit FuzzyMod®

FuzzyMod© ist ein an der TU Ilmenau entwickeltes Tool zur Gewinnung von Regeln aus Lerndaten. Damit wurden die für unterschiedliche Manöversituationen durch einen Operateur in der virtuellen Welt vorgenommenen Steuerhandlungen in Form von WENN-DANN-Regeln generalisiert, so dass sie zukünftig genutzt werden können, um dem Manövermanagement in ähnlichen Situationen in der realen Welt eine autonome Führung des Fahrzeuges zu ermöglichen.

(a) Regelbasierte Modellbildung

Eines der schwierigsten Probleme ist die Ermittlung der Fuzzy-Regeln, die das Prozessverhalten beschreiben. Auf Grund der Komplexität der Prozesse, können die Regeln nicht durch einen Experten formuliert werden. Deshalb ist es sinnvoll, Methoden zur automatischen Wissensermittlung einzusetzen. Dazugehören Cluster-Verfahren, mehrstufige Methoden und die Verwendung von Verfahren des maschinellen Lernens, wie z.B. ID3 von Quinlan [Qui92]. Hier wird gezeigt, wie der ID3-Algorithmus zum induktiven Lernen von Fuzzy-Regeln für statische und dynamische Systeme erweitert und eingesetzt werden kann. Außerdem erfolgt eine Optimierung der Fuzzy-Sets, die wesentlich zu einer Verbesserung der Modellgüte beiträgt. Durch eine iterative Gestaltung dieses Prozesses, wird für eine vorgegebenen Anzahl von Attributen zur Beschreibung der linguistischen Variablen, ein „optimales“ Modell gefunden. Die Fuzzy-Modelle können für verschiedene Zwecke eingesetzt werden.

(b) Methode und Systembeschreibung

Das statistische bzw. dynamische Verhalten von Systemen kann durch folgende Gleichungen beschrieben werden:

$$y = f\,(u_1,u_2,u_3,...,u_n) \tag{6.29}$$

und

$$y(k) = f\,(y(k\text{-}1),...,y(k\text{-}m),u(k\text{-}1),...,u(k\text{-}n)) \tag{6.30}$$

wobei y der Ausgang des Systems ist, u_1, u_2, u_3, …,u_n die Eingänge des statischen MISO-Systems sind, y(k) der abgetastete Ausgang und y(k-1),…, y(k-m), u(k-1),…, u(k-n) die abgetasteten verzögerten Aus- und Eingänge des dynamischen Systems sind und f eine nichtlineare Funktion darstellt.

Die funktionalen Zusammenhänge in Gl.(1) und Gl.(2) können in Form von Regeln formuliert werden, wenn die Ein- und Ausgangsgrößen des Systems durch linguistische Attribute in Abhängigkeit von den gemessenen Werten beschrieben werden. Diese Attribute werden

ermittelt, indem der gesamte Aussteuerbereich der Prozesseingangs- und Ausgangsgrößen zunächst in eine vorgegebene Anzahl von n gleich großen Intervallen eingeteilt wird. Zur Fuzzyfizierung der Prozessgrößen werden den Intervallen dreieckförmige Zugehörigkeitsfunktionen so zugeordnet, dass sie sich an den Intervallgrenzen beim Wert 0,5 schneiden. Die Randintervalle erhalten einseitig offene Zugehörigkeitsfunktionen (Abbildung 6.84). Somit gehört ein Messwert mit unterschiedlichen Zugehörigkeitswerten zu zwei Intervallen. Die Messwerte werden nun entsprechend der festgelegten Intervalle in linguistische Attribute transformiert, so dass eine Beschreibung des statischen bzw. dynamischen Prozessverhaltens durch linguistische Ausdrücke entsteht (Tabelle 6.8).

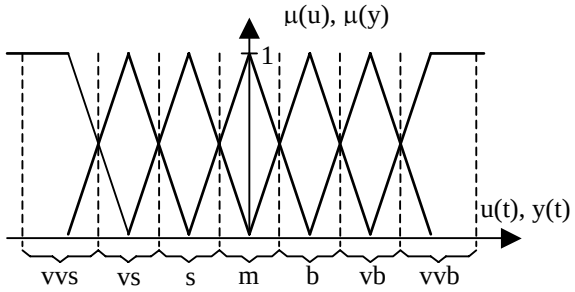


Abbildung 6.84: Fuzzyfizierung der Signale

Static	**y**	**u_1**	**u_2**	**u_3**	**u_4**
Dynamic	**y(k)**	**y(k-1)**	**y(k-2)**	**u(k-1)**	**u(k-2)**
	vvb	vb	b	vb	s
	vvb	vvb	vb	b	vb
	b	vvb	vvb	vb	b
	s	b	vvb	vvb	vb
	.	.	.	.	.
	.	.	.	.	.
	.	.	.	.	.

Tabelle 6.8: Lernbeispiele für den ID3-Algorithmus

Der ID3-Algorithmus [Qui92] generiert aus den linguistischen Beispielen einen optimalen Entscheidungsbaum, aus dem Produktionsregeln der Form

IF (u_1 = vs AND u_3 = b) THEN y = m bzw. IF (y(k-1) = vb AND u(k-2) = s) THEN y(k) = vvb,

gewonnen werden können, die das statische bzw. dynamische Systemverhalten beschreiben. Der Entscheidungsbaum ist so aufgebaut, dass sich die Bedeutung der Attribute für den Entscheidungsprozeß mit zunehmender Tiefe verringert und nicht relevante Attribute unberücksichtigt bleiben.

(c) Generierung des Regelwerks

In Analogie zum klassischen Fuzzyentwurf, wird der datenbasierte Systementwurf durch eine unbekannte Relation zwischen dem Eingang und den Ausgangsgrößen (Regelbasis) gekennzeichnet. Die Erzeugung der Regelbasis basiert auf der Analyse der vorhandenen Signale der Eingangs- und Ausgangsgrößen. In der automatisierten Entwurfsmethode, die hier angewandt wird, wird die Regelbasis mit dem bekannten ID3-Algorithmus von [Qui92], [OW92] erzeugt. In diesem Algorithmus werden die Fuzzy-Sets, die das Signal

kennzeichnen, als Signal- Symbol Transformationen verstanden. Die Rückumwandlung, d.h. Defuzzyfizierung, wird in der entgegengesetzten Richtung durchgeführt.

Die Grundidee des ID3-Algorithmus basiert auf der Erzeugung eines Entscheidungsbaums (Gleichungen (3), (4)). Nachdem man das Verfahren mit einer Primärkonfiguration begonnen hat, werden die erzeugten Fuzzy Sets mit den Regeln verknüpft, und die letzten werden dann auf ihre "Korrektheit" überprüft (mittlerer Informationsgehalt).

Der ID3-Algorithmus umfasst folgende Schritte:

1) Suche des Attributes a_j, für das sich entsprechend dem mittleren Informationsgehalt ein Minimum für die Entropie ergibt:

$$H\left(C/a_j\right) = \sum_{i=1}^{N} P\left(X_i\right) H\left(C/X_i\right) \Rightarrow \min \tag{6.31}$$

dabei ist:

$$H\left(C/X_i\right) = -\sum_{j=1}^{n} P\left(C_j/X_i\right) \log_2 P\left(C_j/X_i\right) \tag{6.32}$$

$H(C/a_j)$ - Entropie des für die Entscheidungen C_j , wenn Eingang a_j vor- liegt

$P(C_j / X_i)$ - bedingte Wahrscheinlichkeit für das Attribut C_j der Ausgangsvariablen y unter der Bedingung, dass eine Klasse X_i von Beispielen mit gleichen Attributen für die Modelleingänge (u_1, u_2, ... bzw. u(k-i), y(k-j) existiert.

n - Anzahl der Attribute der Ausgangsgröße

N - Anzahl äquivalenter Klassen von Beispielen mit gleichen Eingangsattributen

a_j - Modelleingangsgröße mit der geringsten Unbestimmtheit

2) Streichen der in 1. ermittelten Größe und der Beispiele, die in äquivalenten Klassen mit $H(C/X_i) = 0$ enthalten sind.

3) Einsetzen der in 1. gefundenen Größe in den Entscheidungsbaum und Wiederholung der vorgegebenen Schritte, beginnend mit 1. bis entweder alle Eingänge oder alle Beispiele erfasst sind.

4) Ableitung der Regeln aus dem Entscheidungsbaum

Auf Grund der Eigenschaften des ID3-Algorithmus wird ein minimal diskriminierender Entscheidungsbaum für die vorliegenden Beispiele erstellt, so dass nicht alle möglichen Kombinationen der linguistischen Variablen verwendet werden müssen. Die Defuzzyfizierung erfolgt durch Schwerpunktbildung, wobei die einzelnen Anteile durch die Minimum-Inferenz-Methode ermittelt werden. Dieser Prozess kann durch Verwendung von Singletons für die Ausgangsgröße noch vereinfacht werden, ohne die Güte wesentlich zu beeinflussen.

(d) Optimierung der Zugehörigkeitsfunktionen

Da zur Erstellung der Regeln von gleich großen Intervallen und gleichförmigen Zugehörigkeitsfunktionen für alle Attribute ausgegangen wurde, kann man erwarten, dass durch eine günstigere Gestaltung der Zugehörigkeitsfunktionen eine weitere Verbesserung der Fuzzy-Modelle möglich ist. Dazu soll eine Optimierung der Fuzzy-Sets so erfolgen, dass der mittlere quadratische Fehler zwischen Modell- und Systemausgang minimiert wird:

$$Q = \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \Rightarrow \text{Min} \qquad \textbf{(6.33)}$$

mit: y_i - Systemausgang
n - Anzahl der Datensätze
$\hat{y}_i$ - Modellausgang

Zu diesem Zweck werden die Zugehörigkeitsfunktionen der Eingangsmerkmale durch vier Stützstellen beschrieben, so dass beliebige rechteck-, dreieck- bzw. trapezförmige Zugehörigkeitsfunktionen definiert werden können. Als Nebenbedingungen wird beibehalten, dass einem Messwert maximal 2 Attribute zugeordnet werden und die Summe der Zugehörigkeitsfunktionen $\mu(u)$ gleich eins ist Die Parameter dieser Zugehörigkeitsfunktion sind demzufolge durch die Beginn- und Endpunkte der Intervalle voller Zugehörigkeit (Dachpunkte η_1, η_2 ...,η_{2n-2}) gegeben. Die Fußpunkte ergeben sich aus den Dachpunkten der Zugehörigkeitsfunktion der benachbarten Attribute (Abbildung 6.85).

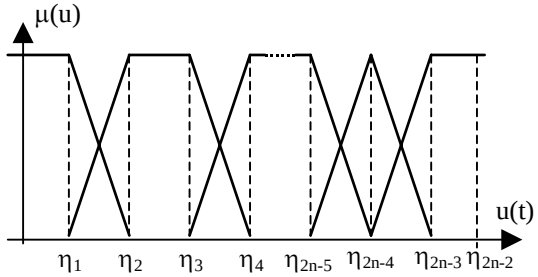


Abbildung 6.85: Zugehörigkeitsfunktionen einer Eingangsgröße

Für ein Fuzzy-Set mit n Zugehörigkeitsfunktionen müssen folglich 2(n-1) Parameter, η_1, η_2 ...,η_{2n-2} optimiert werden. Dabei handelt es sich um ein beschränktes nichtlineares Optimierungsproblem. Zur Lösung werden das Powell-Verfahren [Vac94] sowie ein Optimierungsverfahren mit Evolutionsstrategie nach Schwefel eingesetzt. Nach der Optimierung der Zugehörigkeitsfunktionen ist auf Grund der Verschiebung der ursprünglich gewählten Attributgrenzen die Zuordnung der Messwerte zu den linguistischen Beschreibungen nicht mehr vollständig gültig, so dass die daraus ermittelten Regeln ebenfalls nicht mehr optimal sein müssen. Aus diesem Grund muss der gesamte Vorgang der Reglermittlung und Fuzzy-Set-Optimierung so lange wiederholt werden, bis keine weitere Verbesserung der Güte zu verzeichnen ist. Dabei werden die optimierten Zugehörigkeitsfunktionen des vorhergehenden Schrittes als Start-Zugehörigkeitsfunktionen für den nächsten Lernschritt verwendet. Erfahrungsgemäß reichen 2-4 Iteration aus, um das optimale Fuzzy-Modell zu ermitteln.

Der beschriebene Algorithmus zur Fuzzy-Modellbildung wurde in einem Programmsystem FuzzyMod® realisiert. Zusätzlich wurden die Module zur Fuzzyfizierung, Inferenz und Defuzzyfizierung so miteinander verknüpft, dass eine online Verarbeitung der offline erzeugten Regeln und Zugehörigkeitsfunktionen ermöglicht wird. Diese Funktionen bilden den Fuzzy-Regler, der online eingesetzt wird.

Ein **Fuzzy-Regelwerk** wurde mit dem Tool FuzzyMod© aus den in der virtuellen Welt generierten Daten erstellt. Das Regelwerk besteht aus drei Komponenten, einem Teil zur Steuerung des AUV zum Ziel, einem zweiten zum Ausweichen von Hindernissen auf dem Weg zum Ziel und einem dritten zur Steuerung der Geschwindigkeit. Durch diese Aufteilung verbessert sich das Lernverhalten der Module.

6.3.5.5 Fuzzy Inferenzsystem (FIS) für den Regler

Das erstellte **Fuzzy Inferenzsystem (FIS)** dient zur Abarbeitung der Regeln und zur Steuerung des Unterwasserfahrzeuges in unbekannter Umgebung (Abbildung 6.86). Wie in der Abbildung veranschaulicht wird, verarbeitet das FIS-System drei Regelbasen (Zielansteurung (R_1), Hindernisvermeidung (R_2) und Geschwindigkeitsberechnung (R_3)) zur Berechnung der erforderlichen Steuerkommandos.

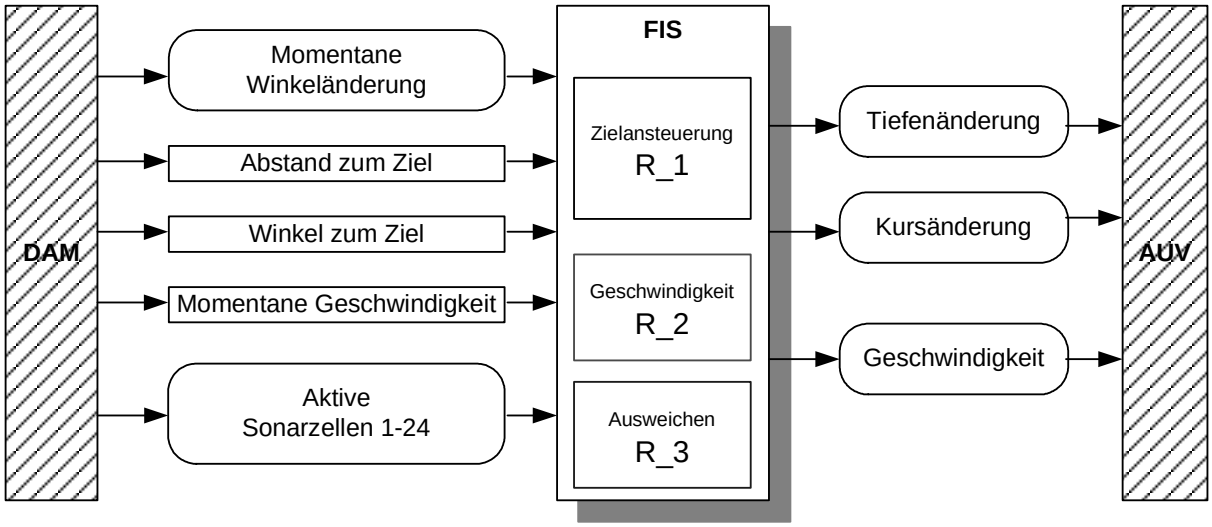
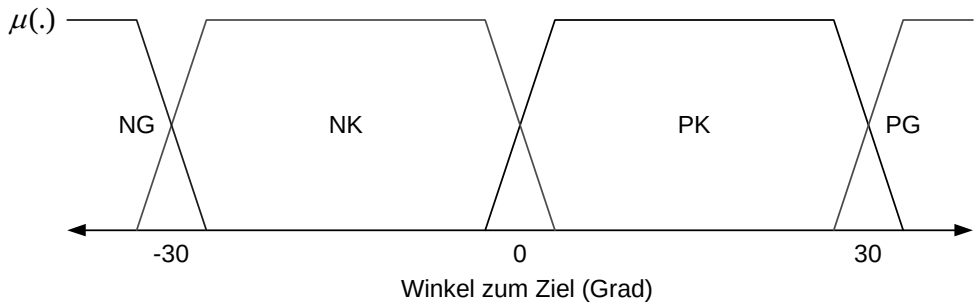


Abbildung 6.86: Fuzzy-Inferenzsystem FIS (mit R-Regelwerk)

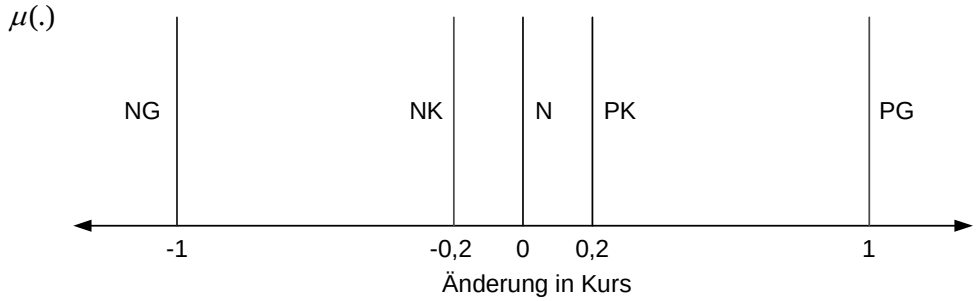
Für das Modul zur Zielansteuerung werden nur zwei Eingangsvariablen angefordert, Zielposition und Abstand zum Ziel. Für das Modul zur Hindernisvermeidung werden nur die Sonarzellen-Abstandswerte als Eingänge angefordert und für das Modul, das für die Berechnung der Geschwindigkeit verantwortlich ist, werden der Abstand zum Hindernis, der Abstand zum Ziel und die vorhergehende Geschwindigkeit als Eingänge angefordert.

Als Beispiel werden die automatisch generierten Zugehörigkeitsfunktionen und die Regeln für das Zielansteuerungsmodul sowie die Geschwindigkeitsberechnung gezeigt. Die Optimierung nach dem 1. Iterationsschritt zeigte noch nicht das bestmögliche Modell. Erst im 4. Schritt ist ein Optimum bezüglich des Gesamtmodells (Regeln+Fuzzy-Sets) erreicht. Dabei kann sich sowohl die Anzahl der Regeln als auch ihre Semantik ändern. Im Abbildung 6.87(a) und (b) sind die optimierten Zugehörigkeitsfunktionen für ausgewählte Eingangs- und Ausgangsgrößen des Zielansteuerungsmoduls dargestellt. Singletons, wie in Abbildung 6.87(a) können für die Ausgänge verwendet werden, ohne die Güte des Modells wesentlich zu beeinflussen. Für das Zielansteuerungsmodul lauten die ersten 3 Regeln z.B.:

```
1. IF ( Input_0 = NG ) THEN (Output := NG);
2. IF ( Input_0 = PK ) THEN (Output := PK );
3. IF ( Input_0 = PG ) THEN (Output := PG );
```



(a) Ausgewählte Eingänge des Zielansteuerungsmoduls (Input_0)



(b) Ausgewählter Ausgang des Zielansteuerungsmoduls

Abbildung 6.87: Optimierte Zugehörigkeitsfunktion für das Zielansteuerungsmodul

Im Bereich von 0 Grad muss fein gesteuert werden, weshalb die Zugehörigkeitsfunktionen des Ausganges (NK, N, PK) eng beieinander liegen. Der zweite Ausgang des Zielansteuerungsmoduls hängt direkt vom Abstand zum Ziel ab. Entweder wird das Fahrzeug gebremst (0 Knoten) oder es fährt mit normaler Geschwindigkeit (3 Knoten). Hierbei sollte man nur den Bremsweg des Unterwasserfahrzeuges beachten. Deshalb sind auch die Regeln einfach, es gibt nur zwei:

```
1. IF ( Input_0 = Term_0_0 ) THEN (Output := Term_0 );
2. IF ( Input_0 = Term_0_1 ) THEN (Output := Term_1 );
```

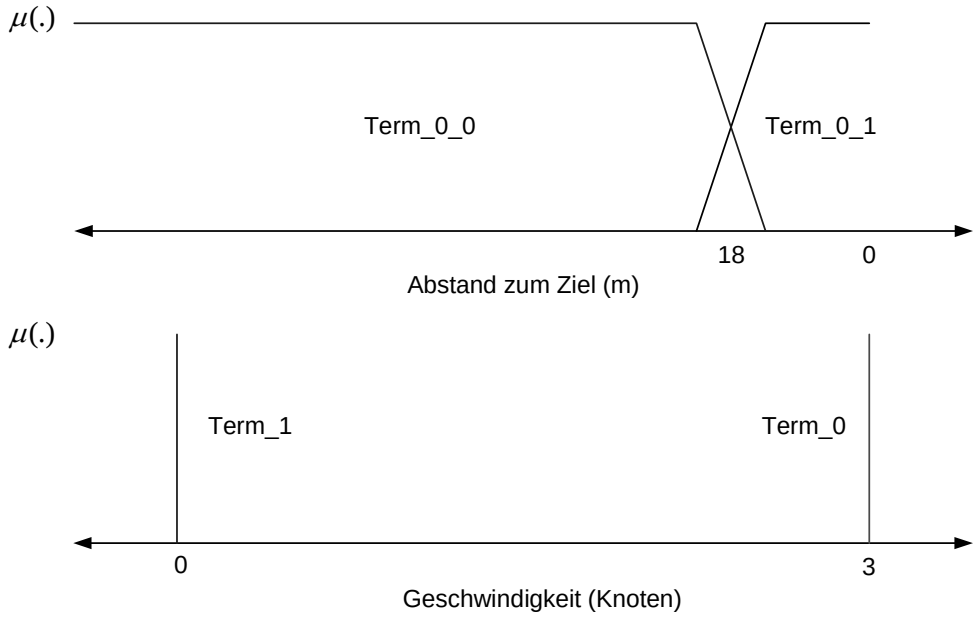
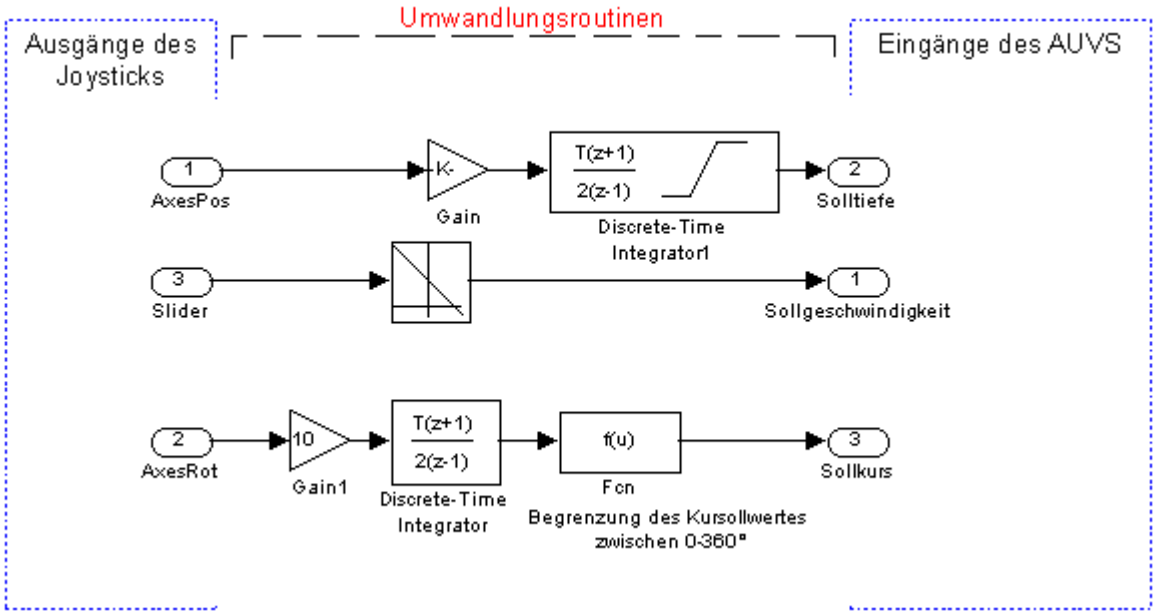


Abbildung 6.88: Optimierte Zugehörigkeitsfunktion für das Zielansteuerungsmodul(v)

In jedem Schritt, in dem ein Hindernis identifiziert wurde, wird das Modul für die Hindernisvermeidung aktiviert. Wird durch die Sonarzellen kein Hindernis registriert, wird die Ansteuerung zum Ziel aktiviert und in jedem Fall die Geschwindigkeit durch das Geschwindigkeits- Modul dementsprechend errechnet. Die Teilung des gesamten Reglers in einfache Module, ermöglicht die Nutzung von vorher bekannten Informationen, wie z.B. wann die einzelnen Module aktiviert werden sollen.

Die zwei Module (Datenaufbereitungsmodul und Fuzzy-Inferenzsystem) wurden in Form von S-Funktionen und BCB6 C++ Funktionen implementiert. Nach dem Lernen und der Validierung wurde anstelle des Joystick (Experte) der Regler benutzt. Die Ergebnisse für die Navigation in verschiedenen, unbekannten repräsentativen Umgebungen sind im folgenden Abschnitt veranschaulicht.

Anzumerken ist, dass beim Ersetzen der Funktionen des Joysticks, die Ausgangswerte zwischen –1 und 1 liegen. Diese werden mit Zusatzfunktionen in Tiefen-, Geschwindigkeits- bzw. Kurswerte umgewandelt (siehe Abbildung 6.89).



mit:

$$f(u) = u[1] - \left(\left(u[1] \geq 0\right) \cdot floor\left(\frac{u[1]}{360}\right) + \left(u[1] < 0\right) \cdot ceil\left(\frac{u[1]}{360} - 1\right) \right) \cdot 360$$

Abbildung 6.89: Umwandlungsroutinen für die Joysticksausgangswerte

6.3.5.6 Technische Umsetzung

Die Lerndaten sollen aus den Testfahrten eines Operateurs in der virtuellen Welt aufgezeichnet werden. Hierbei werden Hindernissituationen vorgegeben, welche der Operateur zu lösen hat.
Zur Durchführung der Fahrten wurde eine Testumgebung unter MATLAB/SIMULINK aufgebaut, welche die nachfolgenden Aufgaben erfüllt:

- Ansteuerung des Fahrzeuges mit Eingabegeräten (Joystick, Lenkrad, ...) unter MATLAB
- Nachbildung des geregelten Fahrzeugverhaltens
- Ansteuerung bzw. Kommunikation mit der virtuellen Realität KISMET [Kis04] der Universität Karlsruhe

- Darstellung der Instrumente und des Sonars zur sicheren Führung des Fahrzeuges
- Aufzeichnung der Fahrdaten (Steuerkommandos der Eingabegeräte, Lage des Fahrzeuges, Position der Hindernisse)

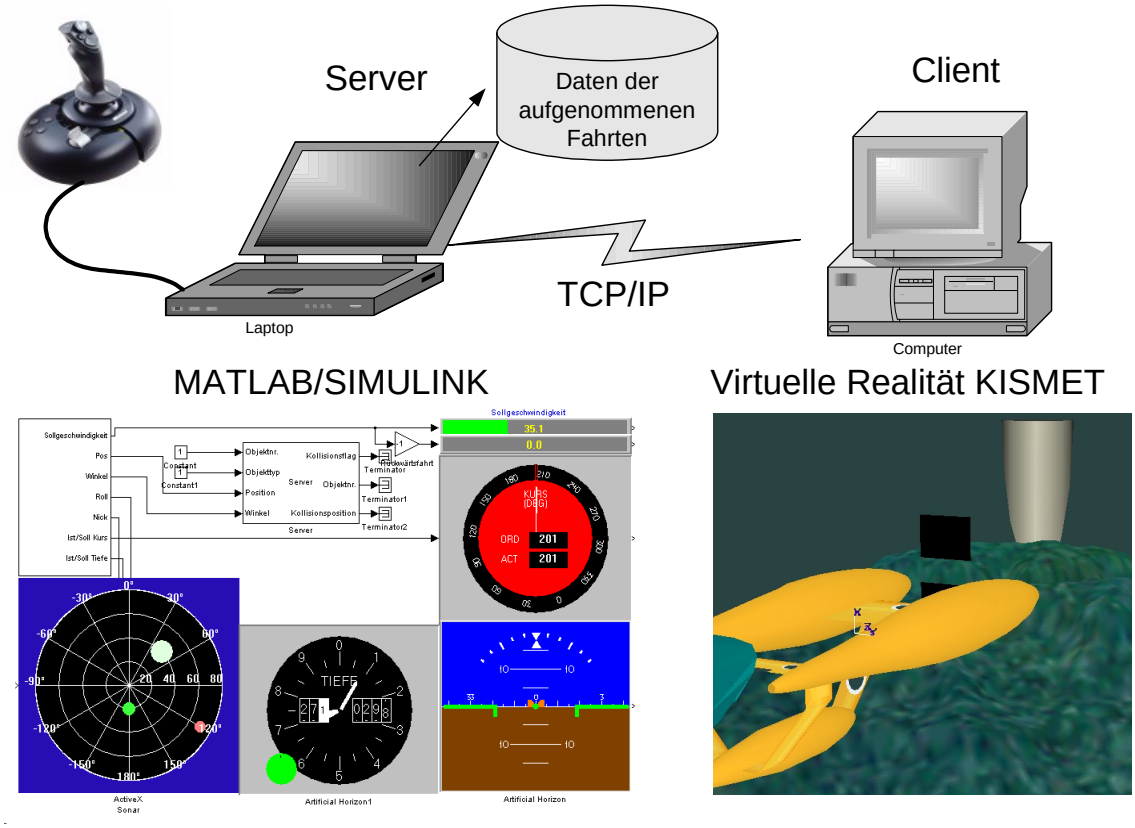


Abbildung 6.90: Umgebung zur Durchführung der Testfahrten in der Virtuellen Realität

Die Testumgebung wurde auf zwei Rechner verteilt. Auf dem ersten Rechner läuft MATLAB welches die o.g. Aufgaben erfüllt. Die rechenintensiven Algorithmen der virtuellen Realität laufen auf einem zweiten Rechner. Abbildung 6.90 zeigt die aufgebaute Testumgebung.
Durch den Aufbau der Testumgebung unter MATLAB/SIMULINK ist eine Modifikation der einzelnen Funktionen leicht möglich. Des Weiteren kann auf vorhandene Komponenten unter MATLAB/SIMULINK zur Erfüllung der o.g. Aufgaben zurückgegriffen werden.
So konnte die Datenaufzeichnung mit der bestehenden SIMULINK-Funktionalität durchgeführt werden. Die Verwendung des „Dials & Gauges Blocksets“ von MATLAB [DGB02] erlaubte die Nutzung vordefinierter Anzeigeinstrumente zur Darstellung der aktuellen Fahrzeuglage.
Das Modell des geregelten Fahrzeugverhaltens wurde durch eine Auswertung (durchgeführte AUV – Simulationen unter Excel bei STN-Atlas) der simulierten Fahrtrajektorien und den Anforderungen an das Fahrzeug von „Mission Level Design“ mit dem Designtool MLDesigner erstellt. Dieses Modell ist durch MLD in ein MATLAB - konformes Modell umgesetzt worden, welches Bestandteil dieser Testumgebung ist.
Die Ansteuerung/Abfrage der Eingabegeräte erfolgte durch s-Functions unter Verwendung von DirectX / DirectInput [DiX04]. So ist es möglich, direkt auf die Hardware zuzugreifen und Geräte, welche eine Force Feedbackunterstützung besitzen, anzusteuern.
Der Datenaustausch zwischen der SIMULINK-Testumgebung und der VR KISMET erfolgt über ein TCP/IP Protokoll.
Zu diesem Zweck wurde eine s-Function geschrieben, welche eine Socketverbindung aufbaut und die Daten von SIMULINK in ein Datentelegramm umwandelt und zur Virtuellen Realität sendet. Die empfangenen Daten werden in einem Programm ausgelesen und an KISMET über Shared Memory übermittelt.
Bei den Testfahrten zeigte es sich, dass bei einer reinen visuellen Erfassung der Umgebung ein sicheres Ausweichen und Umfahren der Hindernisse nicht möglich ist (begrenztes

Sichtfeld). Aus diesem Grund wurde ein zusätzliches Anzeigeinstrument in Form eines Sonarabbildes für die aufgefassten Hindernisse in Fahrzeugnähe entwickelt. Diese Anzeige ist als ActiveX-Steuerelement programmiert, wodurch eine leichte Einbindung in SIMULINK möglich ist. Abbildung 6.91 zeigt die Blöcke der Testumgebung in der SIMULINK - Bibliothek.

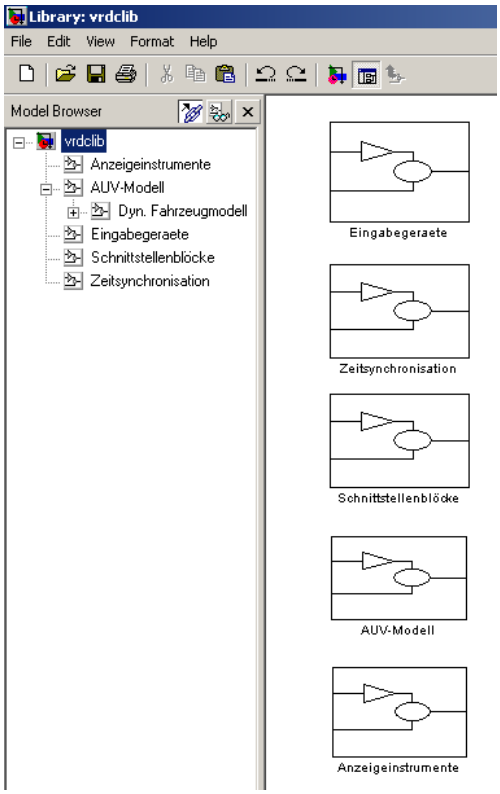
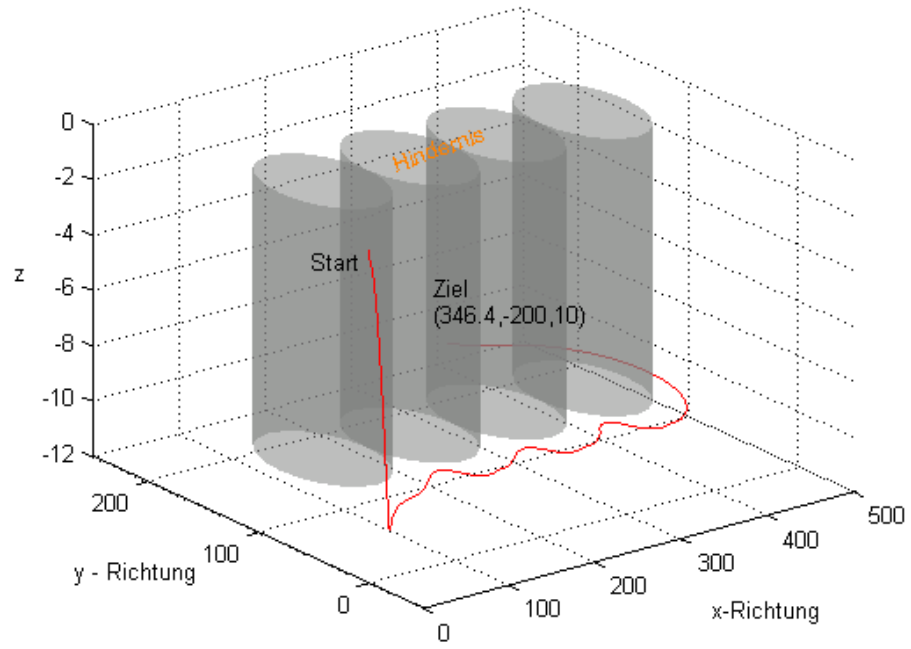


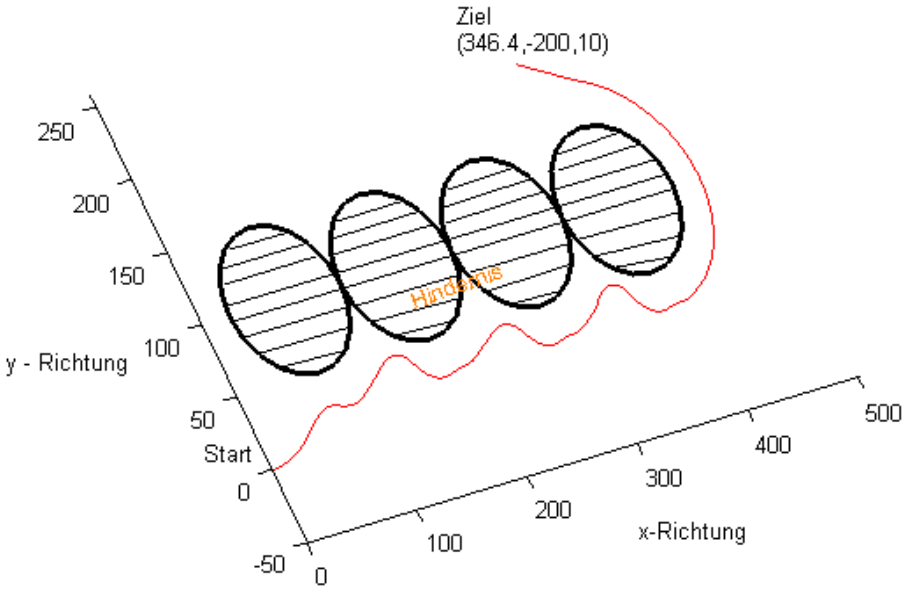
Abbildung 6.91: SIMULINK-Bibliothek

6.3.6 Ergebnisse der simulativen Untersuchungen

Die Ergebnisse mit dem durch die TU Ilmenau für Matlab® erstellten AUV-Modell für ausgewählte Umweltszenarien, zeigt die Abbildung 6.92 - Abbildung 6.95. Die aus dem Regelsystem ermittelten Fahrtrajektorien zeigen das reaktive Ausweichen des AUV bei Hindernissen und bei der Zielansteuerung. Die Positionierung der Hindernisse wurde so vorgenommen, dass sie für eine Vielzahl von in der Praxis auftretenden Situationen möglichst repräsentativ ist.

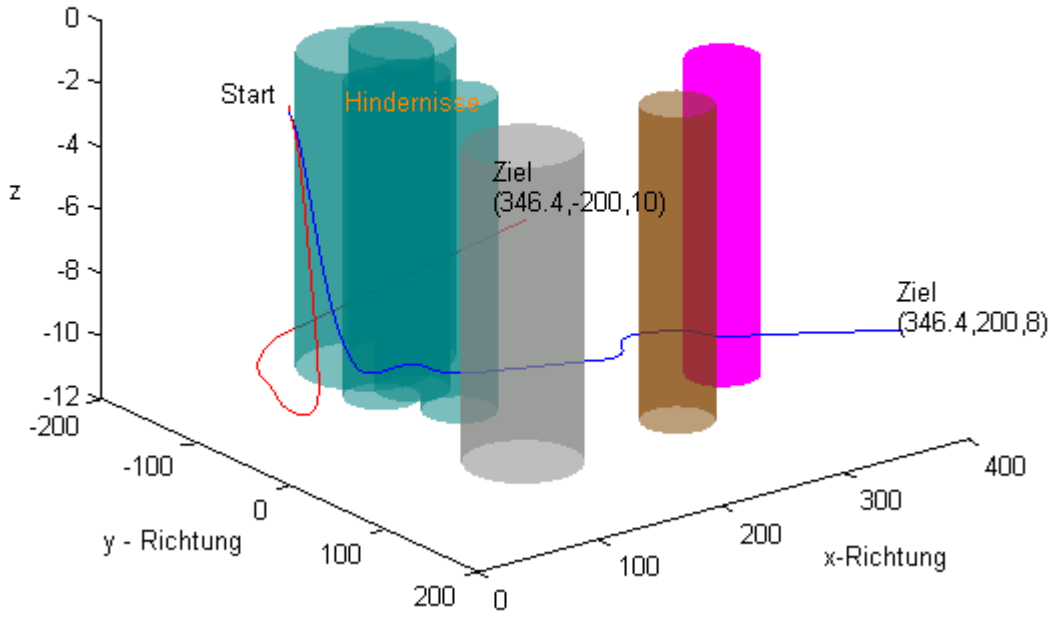


(a) 3D Ansicht

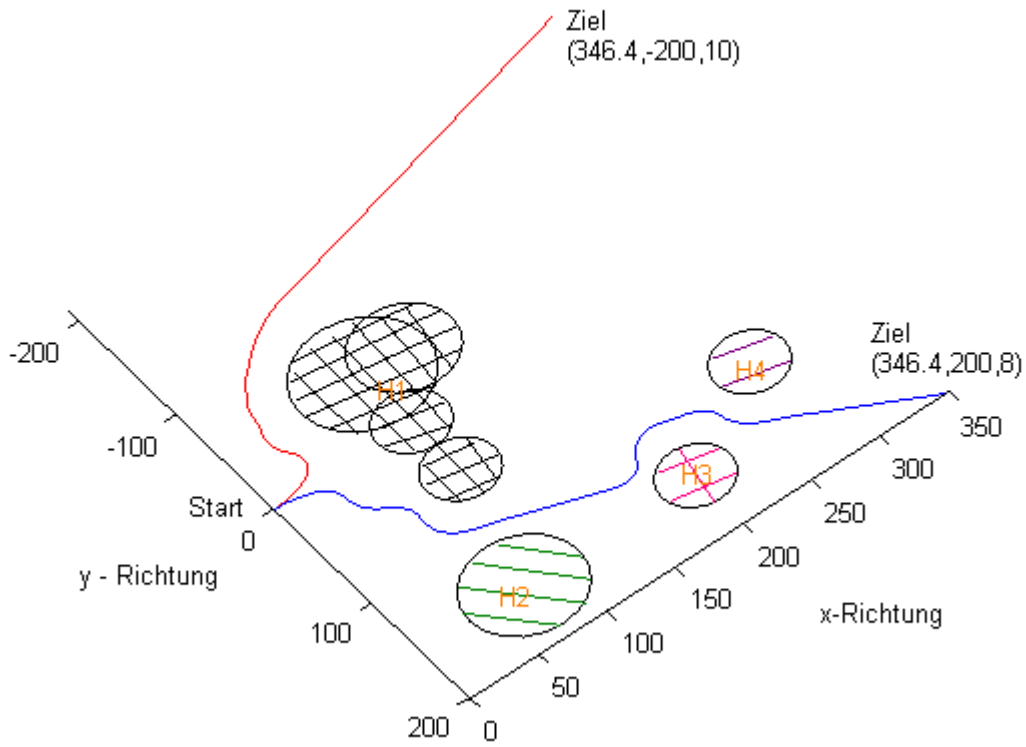


(b) Ansicht von Oben

Abbildung 6.92: Wandverfolgung mit Reiz - Reaktion - Regeln

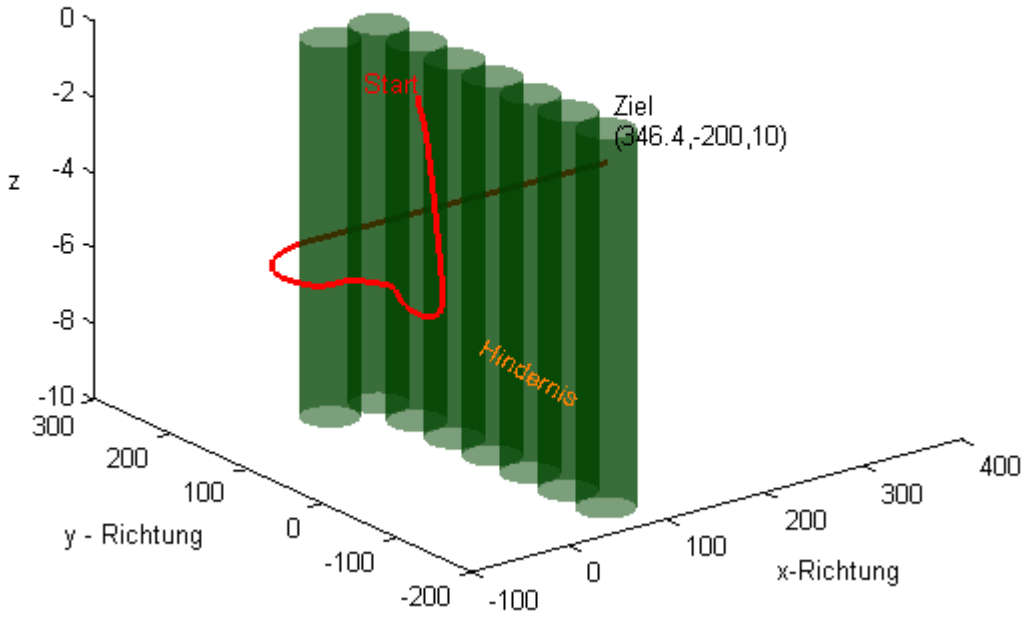


(a) 3D Ansicht



(b) Ansicht von Oben

Abbildung 6.93: Ansteuerung von bzw. zu verschiedenen Start – und Zielpunkten



(a)

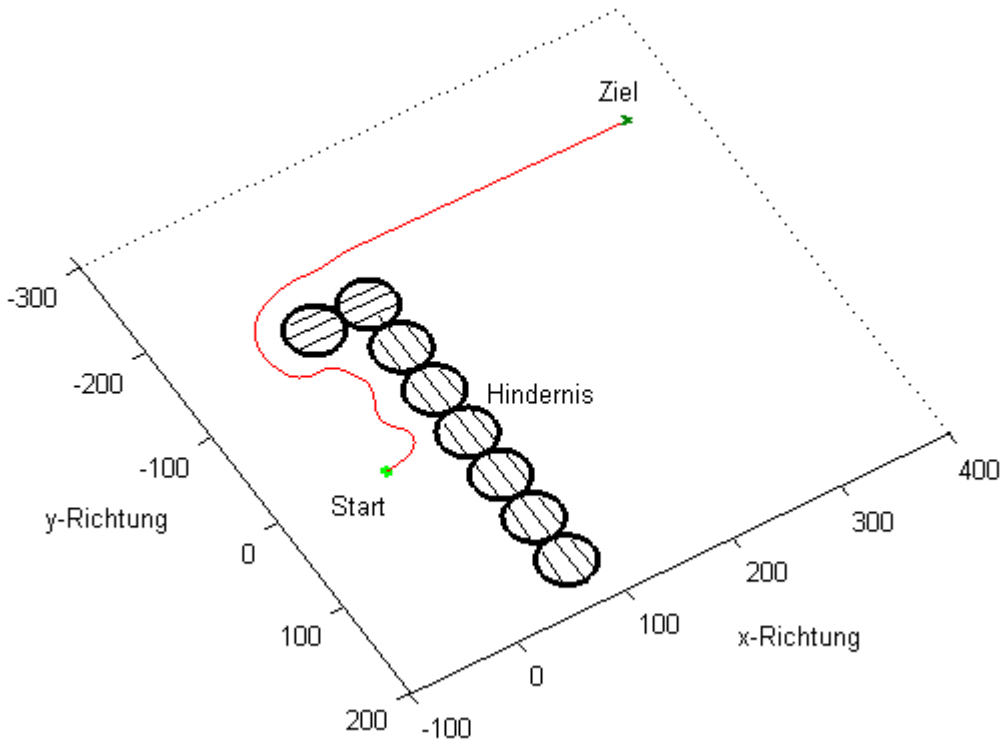
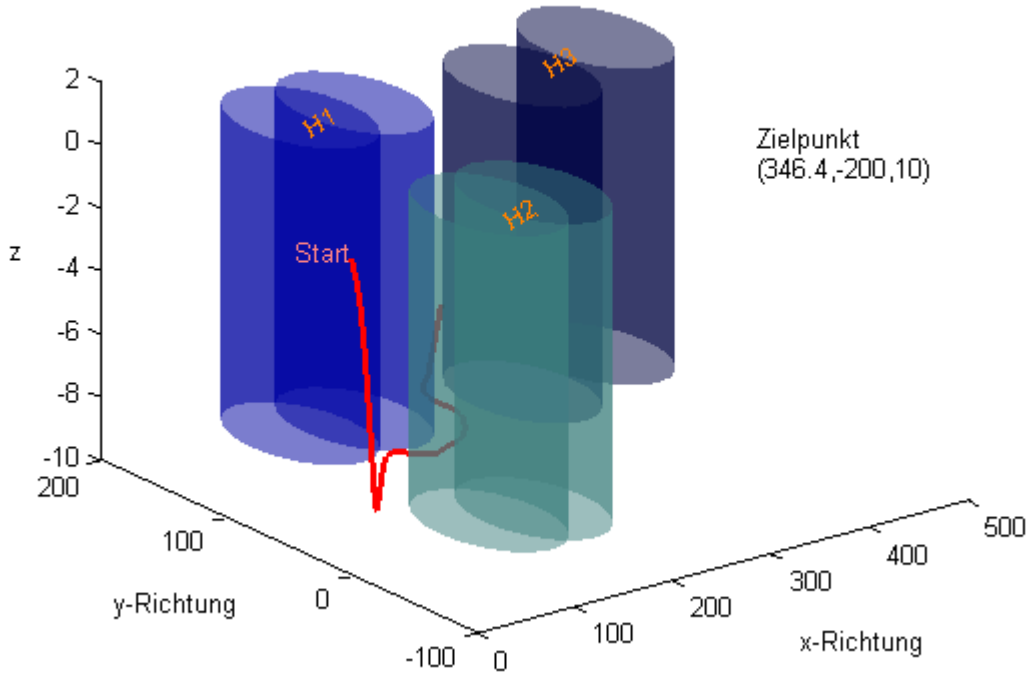


Abbildung 6.94: AUV macht eine ca. – 180 Umdrehung um das Ziel zu erreichen



(a) 3D - Ansicht

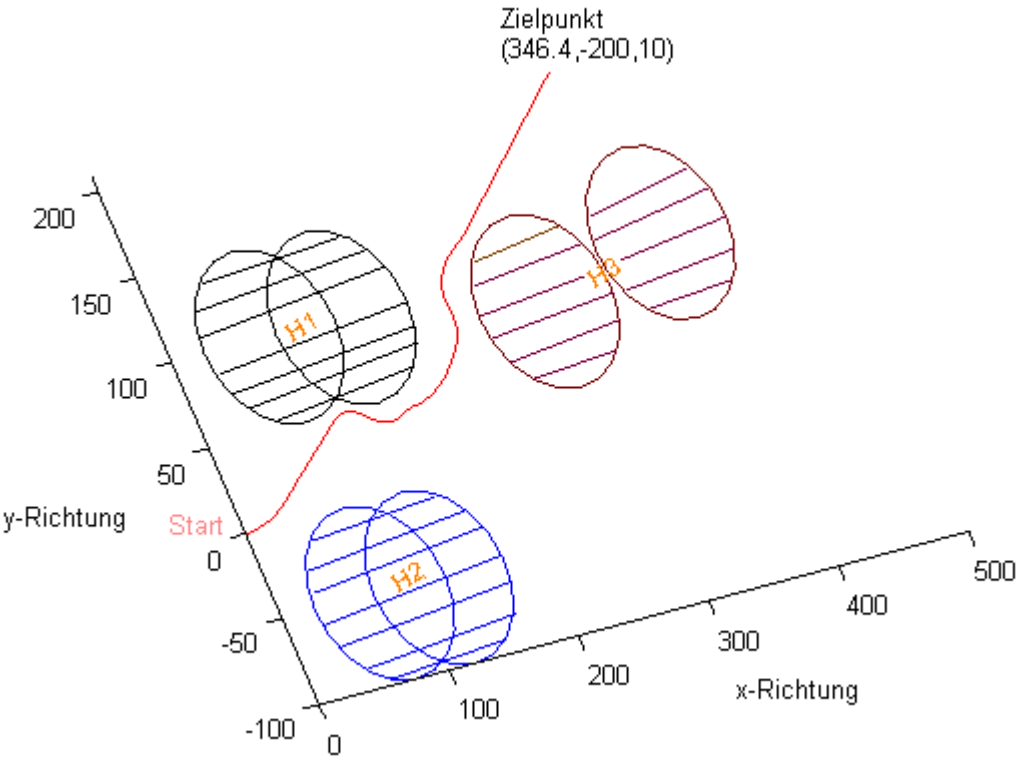


Abbildung 6.95: Slalom durch Hindernisse zum Ziel

Das Fuzzysystem für das vorhandene Matlab – Modell des Unterwasserfahrzeuges ist fertiggestellt. Alle Funktionen sind in Form von BCB6 Funktionen implementiert. Die Funktionen werden in das Ausweichmodul der AUV-Software integriert und haben keine direkte Schnittstelle zu den restlichen Softwarekomponenten des Unterwasserfahrzeuges.

6.3.7 Dateien und Verzeichnisse

Das Fuzzy-Inferenzsystem von FuzzyMod® erfordert, dass sich die Dateien der Regelbasis mit der Endung *.fuz,* die Datei des Arbeitsbereiches (enthält alle Variablen) mit der Endung *.fpj* und die Umgebungsdatei mit der Endung *.evr* in einem gemeinsamen Verzeichnis befinden (Abbildung 6.96). Alle von FuzzyMod® erzeugten Dateien haben die gleichen Namen mit einer individuellen Erweiterung für ihre Funktionen. Für die drei Module, Zielansteuerung, Hindernisvermeidung und Geschwindigkeitsberechnung befinden sich die Dateien in den Verzeichnissen go2goool01, Turn2goool01 bzw. vel2goool01.

Abbildung 6.96: Verzeichnisstruktur der automatisch erzeugten Dateien

6.3.8 Zusammenfassung der Ergebnisse

Zur programmtechnischen Umsetzung des Lernmoduls des Fahrzeuges wurden alle für das Lernen und die Manöverdurchführung notwendigen Parametern erfasst und zusammengestellt. Die Eingabeparameter des Lernmoduls beinhalten die Objektdaten, den Zustand des Fahrzeuges, Umweltinformationen, Sollbahninformationen und alle Daten, die zur Ausführung von Fahrmanövern intern generiert werden (Manöverkommandos).

Folgende Aufgaben wurden abgeschlossen:

a) **FuzzyMod©** : Ein an der TU Ilmenau entwickeltes Tool zur Gewinnung von Regeln aus Lerndaten. Damit wurden die für unterschiedliche Manöversituationen durch einen Operateur in der virtuellen Welt vorgenommenen Steuerhandlungen in Form von WENN-DANN-Regeln generalisiert, so dass sie zukünftig genutzt werden können, um dem Manövermanagement in ähnlichen Situationen in der realen Welt eine autonome Führung des Fahrzeuges zu ermöglichen.

b) **Datenaufbereitungsmodul** (DAM): Dieses Modul dient der Umwandlung von absoluten Fahrzeugdaten wie Absolutposition, -geschwindigkeit, -sonardaten in relative Daten wie den Winkel und den Abstand zum Ziel oder den Winkel und den Abstand zum Hindernis. Dieses Modul wurde in das Ausweichmodul integriert.

c) **Lerndaten** (Steuerhandlungen, die durch einen Experten bei der Steuerung des Unterwasserfahrzeuges erzeugt werden) wurden für ausgewählte Szenarien der Hindernisvermeidung in der virtuellen Umgebung mit dem Matlab-Modell, das durch die TU Ilmenau erstellt wurde und mit dem das dynamische Verhalten des Fahrzeuges unter Wasser nachgebildet wird, erfasst. Diese Daten dienen als Grundlage des Lernvorganges.

d) Ein **Fuzzy-Regelwerk** wurde mit dem Tool FuzzyMod© aus den in der virtuellen Welt generierten Daten erstellt. Das Regelwerk besteht aus drei Komponenten, einem Teil zur Steuerung des AUV zum Ziel, einem zweiten zum Ausweichen von Hindernissen auf dem Weg zum Ziel und einem dritten zur Steuerung der Geschwindigkeit. Durch diese Aufteilung verbessert sich das Lernverhalten der Module.

e) Erstellung eines **Fuzzy Inferenzsystems (FIS)** zur Abarbeitung der Regeln und zur Steuerung des AUV in unbekannter Umgebung .

Die zwei Module (Datenaufbereitungsmodul und Fuzzy-Inferenzsystem) wurden in Form von S-Funktionen und BCB6 C++ Funktionen implementiert. Die Simulation mit dem AUV-Modell, das durch die TU Ilmenau für Matlab erstellt wurde, zeigt zufriedenstellende Ergebnisse für ausgewählte Umweltszenarien. Die aus dem Regelsystem ermittelten Fahrtrajektorien zeigen das reaktive Ausweichen des AUV bei Hindernissen. Die Positionierung der Hindernisse wurde so vorgenommen, dass sie für eine Vielzahl von in der Praxis auftretenden Situationen möglichst repräsentativ ist.
Das Fuzzysystem für das AUV ist fertig gestellt. Alle Funktionen sind in Form von BCB6 – C++ Funktionen implementiert.

6.3.9 Literatur

[DGB02] Dials & Gauges Blockset, User’s Guide, MathWorks 2002

[DKO97] Dung, L.T.; M. Koch; and P. Otto. 1997. „FuzzyOpt – ein Werkzeug zum Entwurf optimaler Fuzzy-Systeme.“*at – Automatisierungstechnik* 45 (1997) 11, 555-556, R. Oldenbourg Verlag.

[DiX04] http://www.microsoft.com/windows/directx/default.aspx

[GG90] Gordon, D. F. and Grefenstette J. J. (1990). Explanations of empirically derived reactive plans. Proceedings of the seventh International Conference on Machine Learning. Austin, TX: Morgan Kaufmann

[Kis04] http://www-kismet.iai.fzk.de

[KO02] Karimanzira D. and Otto, P. (2002). A Self-tuning predictive controller based on instantaneous linearization using neural networks, IASTED International Conference Artificial and Computational Intelligence (ACI 2002), Japan September 25-27.

[KO04] Karimanzira D. and Otto, P. (2004). *A Manoeuvre Management System Based On Machine Learning For An Autonomous Underwater Vehicle*, IWK Tu – Ilmenau, 27-30.

[OW02] Otto, P. ; Wernstedt, J (2002). Fuzzy methods for the optimal emulation of expert decision behaviour and for the optimal controller design. East-West Fuzzy Colloquium 2002, 10th Zittau Fuzzy Colloquium 2002, September 4-6, Proceedings pp. 351-367.

[Qui92] Quinlan, J.R. 1992. *C 4.5. Programs for Machine Learning.* Morgan Kaufmann Publishers San Mateo, California.

[RS90] Ramsey, C. L., Alan C. Schultz (1990). Simulation-assisted learning competition: Effects of noise differences between training model and target environment. Proceedings of the seventh International Conference on Machine learning. Austin, TX: Morgan Kaufmann. pp. 211-215.

[Vac94] Vachkov, G. 1994. “ *Mulitlevel Fuzzy Rule Based Modelling*”. In Proceedings of the Second European Congress on Intelligent Techniques and Soft Computing, Vol. 1, 240-245. EUFIT `94, Aachen.

6.4 Mission Level Design

6.4.1 Mission Level Design für mobile automatische Systeme

6.4.1.1 Mission Level Design

Das Mission Level Design ist eine Entwurfsmethode für komplexe Systeme, die von Schorcht für einen anforderungsorientierten Entwurf von Mobilkommunikationssystemen entwickelt wurde. Die an ein System gestellten Anforderungen werden hier in Form von definierten, typischen Einsatzszenarien, den sogenannten Missionen, beschrieben. Diese dienen der Validierung der Spezifikation indem sie charakteristische Nutzeranforderungen festhalten. Durch die damit verbundene weitere Formalisierung des Entwurfs verschiebt sich der notwendige Arbeitsaufwand in die frühen Entwurfsphasen. Hierdurch soll die Systemintegration weniger aufwendig und vor allem fehlerfrei verlaufen. Für die Simulation wird - soweit möglich - auf verifizierte Bibliotheken zurückgegriffen. Der hiermit verbundene Vorteil liegt in der Wiederverwendung bereits getesteter Bestandteile.

Die Einführung von Missionen ermöglicht eine deutlich verbesserte Interpretation der gewonnenen Ergebnisse. Das Ergebnis ihrer Verwendung zum Test des Systementwurfs sind nicht allgemeingültige Aussagen über das allgemeine Systemverhalten: Durch den Test typischer Fälle werden mittlere, typische Aussagen für Nutzungsszenarien getroffen. Dies ist eine gewünschte Einschränkung: Bedingt durch die den Systemen zugrunde liegende Komplexität sind allgemeingültige Aussagen für alle denkbaren Fälle nicht mehr möglich. Der Missionsauswahl kommt insofern eine erhebliche Bedeutung für die Testergebnisse zu. Ziel muss es sein, eine Auswahl sinnvoller, typischer Varianten des gesamten Einsatzspektrums vorzunehmen. Die erzielten Ergebnisse basieren dabei stets auf der Auswertung der Gesamtheit aller Missionen.

6.4.1.2 Anpassungen für mobile automatische Systeme

Da das Mission Level Design ursprünglich für den Entwurf von Mobilkommunikationssystemen vorgestellt wurde, muss zuerst geklärt werden, welche Anpassungen sich für den Entwurf mobiler automatischer Systeme - wie dem DeepC Fahrzeug - ergeben.

Bei einem mobilen automatischen System handelt es sich um ein technisches und/oder biologisches System, dass sich für einen gewissen Zeitraum selbständig bewegen und Aufgaben erfüllen kann. Es besitzt Sensoren zur Erfassung der eigenen Zustände und/oder der Umwelt. Informationssysteme verbinden diese mit Entscheidungssystemen und Aktoren. Die Aktoren des mobilen Systems garantieren die Bewegung im Raum und übernehmen weitere Aufgaben. Das Entscheidungssystem muss das System auf Basis dieser Sensordaten im Raum selbständig bewegen und es zur Lösung seiner Aufgaben befähigen.

Im Zusammenhang mit mobilen automatischen Systemen ergeben sich an dieser Stelle für das Mission Level Design spezifische Anpassungen.

1. Die ursprüngliche Mission Level Design-Idee eines generellen Mappings verschiedener Architekturen auf ein Funktionsmodell steht im Widerspruch zur Idee einer testweisen Umsetzung der Spezifikation. Eine klar voneinander abgegrenzte Modellierung von Architektur und Funktionalität fällt zudem schwer, wenn die Architektur Funktion übernimmt. Aus diesem Grund vereint der in dieser Arbeit verfolgte Ansatz das Funktions- und das Architekturmodell im Systemmodell. Es enthält beide Aspekte und repräsentiert damit diesen Teil der Spezifikation des Systems.
2. Die Mobilität der Systeme erfordert ein Umgebungsmodell, das die Bewegung des Systems berücksichtigt. Je nach Einsatzzweck ist es dem System zusätzlich möglich, seine Umgebung aktiv zu verändern. Dieser Gesichtspunkt muss auch in die

Modellierung übernommen werden. Dies kann verdeutlicht werden, indem die Rückwirkung des Systemmodells auf das Umgebungsmodell in die Darstellung integriert wird. Das bedeutet aber auch, dass das Umgebungsmodell quasi eine virtuelle Umwelt bereitstellen muss. Daneben stehen mobile automatische Systeme in einer Interaktion mit ihrer Umwelt. Hier ist das Umgebungsmodell nicht mehr durch z.B. Stimulibeschreibungen ([Sch00, S. 22]) modellierbar.
Um diese Anforderung an das Umgebungsmodell zu unterstreichen, wird es im Folgenden als Umwelt bezeichnet.

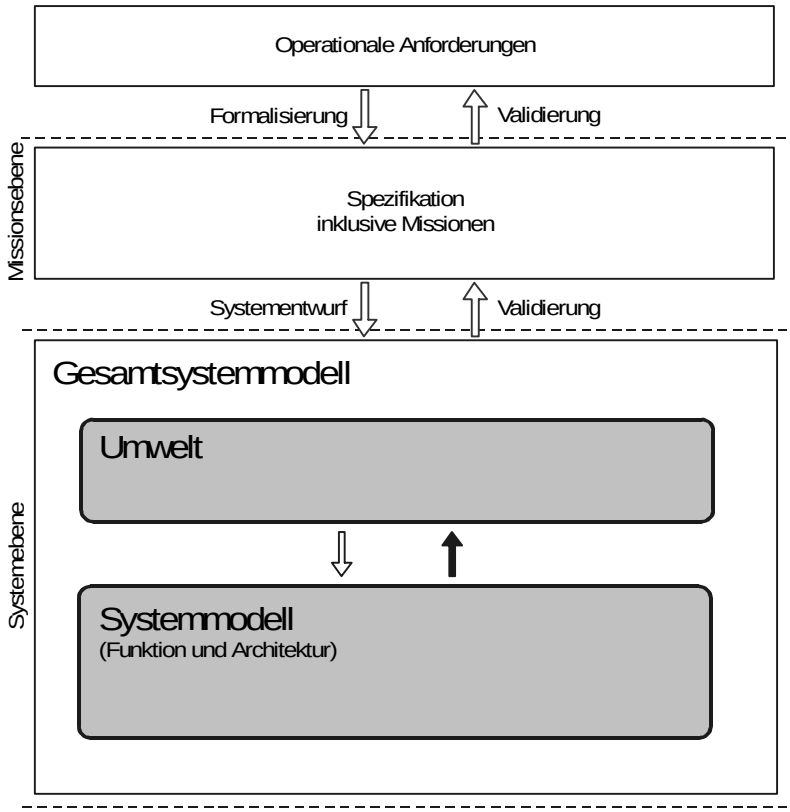


Abbildung 6.97: Missionsbezogener modellgestützter Entwurf für automatische Systeme

Abbildung 6.97 fasst die im Rahmen dieser Arbeit vorgenommenen Änderungen am von Schorcht formulierten Ansatz des Mission Level Designs zusammen. Im oberen Bereich sind die operationalen Anforderungen angeordnet. Sie werden durch die Missionen auf der Missionsebene spezifiziert und durch die Simulation validiert. Die Missionen selbst wirken auf die Systemebene, die vom Gesamtsystemmodell gebildet wird. Das Architektur- und das Funktionsmodell sind nun im Systemmodell vereinigt, welches zusammen mit der Umwelt (früher Umgebungsmodell) das Gesamtsystemmodell konstituiert. Die mögliche Rückwirkung des Systems auf die Umwelt wird durch einen Pfeil zwischen Systemmodell und Umwelt repräsentiert.

6.4.1.3 Konzept für die Durchführung der Simulation

Nachdem die Anpassungen für mobile automatische Systeme diskutiert wurden, wird nachfolgend untersucht, inwieweit die Einführung von Missionen den Simulationsablauf beeinflusst.

Dieser gliedert sich allgemein in drei Phasen: die Modellierung, die Modellnutzung und die Interpretation/Auswertung. Für die Simulation im Rahmen des Mission Level Designs ergeben sich in diesem Zusammenhang zwei Probleme:

- Erstens besteht ein wesentlicher Unterschied im Vergleich zu einer herkömmlichen Simulation darin, dass ein variierendes Gesamtsystemmodell mit unterschiedlichen, festgeschriebenen Missionen parametrisiert werden muss. Für eine große Anzahl von Missionen ist es nicht praktikabel, das Modell vor jeder Simulation manuell zu parametrisieren. Dieses Vorgehen wäre zu arbeitsaufwendig und zudem stark fehleranfällig.

- Zweitens führt die Simulation diverser Missionen mittels ein- und desselben Modells dazu, dass für alle nötigen Ausgabedaten entsprechende Ausgabemöglichkeiten im Modell vorgesehen werden müssen. Da jedoch nicht alle Ausgaben für die Auswertung jeder Mission von Bedeutung sind, ist eine Selektion der Daten notwendig, die in direktem Zusammenhang mit den Missionen steht. Die Konfiguration der Selektion - aber auch die der Visualisierung - ist dabei missionsabhängig (missionsbedingt), aber konstant über die verschiedenen Simulationsläufe einer Mission.

Zur Lösung der beschriebenen Probleme wird ein erweitertes, an das Mission Level Design mobiler automatischer Systeme angepasstes Konzept für die Durchführung der Simulation vorgestellt. Dieses besitzt eine neue Phase der Missionshandhabung in der zentral die Missionserstellung und -verwaltung erfolgt. Das Modell wird auf diese Weise vor jeder Simulation automatisch mit einer Mission konfiguriert, wodurch sich die Arbeit mit den Missionen erheblich beschleunigen und die Fehleranfälligkeit minimieren lässt. Zudem entspricht eine zentrale Handhabung dem Mission Level Design-Prinzip, nach welchem die Missionen einen Teil der zentral festzuschreibenden Spezifikation darstellen. Des weiteren fordert das Konzept für die Auswertung eine erweiterte Funktionalität, damit diese den spezifischen Anforderungen des Umgangs mit Missionen gerecht wird.

Damit besteht die Simulation, wie in Abbildung 6.98 ersichtlich, nunmehr aus vier Phasen, nämlich - wie in herkömmlichen Simulationen üblich - aus der Modellierung, der Modellnutzung und der Auswertung, die um die neue Phase der Missionshandhabung ergänzt werden. Die Auswertung erfährt eine Erweiterung, indem sie missionsspezifisch erfolgt.

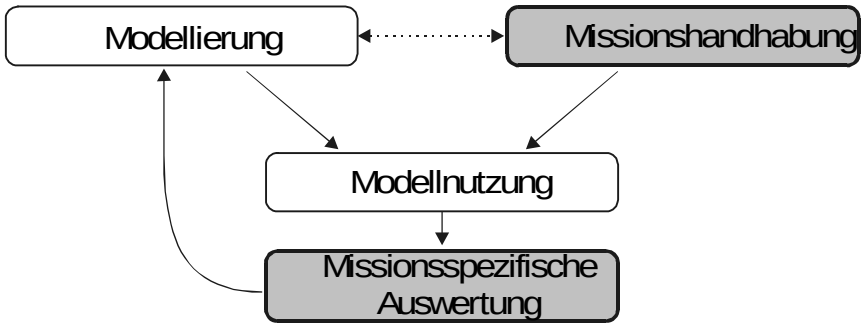


Abbildung 6.98: Phasen der Durchführung des Mission Level Designs

Eine umfassende Charakterisierung der einzelnen Phasen und ihrer Zusammenhänge liefert die folgende Auflistung:

- **Modellierung:**
  Den ersten Punkt der Simulation bildet weiterhin die Modellierung, die die Erstellung des Gesamtsystemmodells mit allen wichtigen Aspekten des Systems beinhaltet und eine Nachbildung der aktuellen Spezifikation des Systems und der Umwelt, in der es agiert, vornimmt.
- **Missionshandhabung:**
  Die Missionen werden - als Teil der Spezifikation - in der Missionshandhabung einmalig festgelegt, später müssen sie lediglich verwaltet werden. Die Festlegung umfasst dabei

(abstrakt formuliert) die Schaffung verschiedener Ausprägungen für die existierenden Missionsparameter oder die Zuordnung der Missionsparameter zu allgemein definierten Missionen. Im Hinblick auf das Mission Level Design besteht die Aufgabe in der Entwicklung einer einfachen und handhabbaren Möglichkeit zur Arbeit mit verschiedenen Missionen.

- **Modellnutzung:**
  In der Modellnutzung werden die Missionen auf das Modell angewendet, wodurch konfigurierte Modelle entstehen. Deren Simulation erfolgt rechentechnisch und erzeugt eine Vielzahl von Simulationsergebnissen. Dabei wird der virtuelle Prototyp in seiner virtuellen Umgebung mit den einzelnen Nutzungsszenarien (Missionen) konfrontiert, wobei sein Verhalten und seine Zustände aufgezeichnet werden.
- **Missionsspezifische Auswertung:**
  Im Rahmen der missionsspezifischen Auswertung werden die Simulationsergebnisse untersucht. Die große Menge anfallender Informationen muss dazu anschaulich aufbereitet werden, was neben der Selektion eine vielseitige Datenaufbereitung erfordert. Des weiteren ist eine Visualisierung nötig, die unterschiedliche Daten jeweils optimal darstellt. Für die Auswertung werden alle relevanten Ergebnisse missionsweise analysiert und daraus eine Gesamteinschätzung des Systemdesigns bezüglich der Nutzeranforderungen erarbeitet. Auf dieser Analyse aufbauend kann im Anschluss der Entwurf fortgeführt und optimiert werden.

Das hier vorgestellte, wesentlich erweiterte Konzept trägt den Besonderheiten von Simulationen im Rahmen des Mission Level Designs Rechnung. Es ergänzt die herkömmlichen Phasen einer Simulation um die neue Phase der Missionshandhabung und um erweiterte Auswertungsmöglichkeiten für Missionen.

6.4.2 Durchführung der Simulation

6.4.2.1 MLDesigner

Ein wesentlicher Bestandteil dieser Arbeit besteht in der Durchführung der Simulationen im Rahmen des Mission Level Designs. Dafür ist die Nutzung des Entwurfswerkzeugs MLDesigner zweckdienlich, da es sich bei diesem um eine speziell auf die Erfordernisse des Mission Level Designs ausgerichtete Anwendung handelt. Des weiteren existierten bei Beginn dieser Arbeit bereits eine Reihe von Projekten, die mit Hilfe von MLDesigner erfolgreich analytische Betrachtungen von Systemen durchführten.

MLDesigner ist eine Simulationsumgebung der vierten Generation, die für die Arbeit im Rahmen des Systementwurfs entwickelt wird. Entsprechend unterstützt das Programm unterschiedliche Abstraktionsebenen für den Entwurf. Sie reichen von Algorithmen und Funktionen über Architektur und Performance (System Level Design) bis zu operationalen Vorgaben/Missionen (Mission Level Design). Die Fähigkeiten des Programms eignen sich dabei für den top-down und den iterativen Systementwurfsprozess. Die folgende Auflistung gibt einen kurzen Überblick der Hauptmerkmale des Programms wieder:

- **Integrierte Entwicklungsumgebung (IDE):**
  MLDesigner bietet eine moderne integrierte Entwicklungsumgebung (siehe Abbildung 1.3), die eine graphische, blockorientierte Modellierung ermöglicht. Diese setzt sich aus unterschiedlichen Bestandteilen (Modelleditor, Filemanager, Modelleigenschafteneditor, ...) zusammen.
- **Domänen (multi):**
  MLDesigner unterstützt verschiedene primäre und sekundäre Modellierungsdomänen. Unter ihnen sind DE, CTDE, DDF und SDF sowie als Subdomänen FSM und HOF. Die CTDE-Domäne ermöglicht beispielsweise analoge und gemischt analogdiskrete Entwürfe. Die einzelnen Domänen lassen sich dabei gemischt verwenden (hybride Simulation).

- **Kompatibilität und Erweiterbarkeit:**
  Um vorhandene Modelle weiterverwenden zu können, wird der Import aus BONeS und COSSAP unterstützt. MLDesigner ist mittels C++ und einer eigenen Beschreibungssprache beliebig erweiterbar.
- **Externe Simulation:**
  Simulationen lassen sich auch außerhalb des Programms durchführen. Dazu wird das Modell übersetzt und mit den Programmbibliotheken verlinkt. Diese externe Abarbeitung beschleunigt die Simulation, wobei die Modellparameter über Parameterdateien weiterhin änderbar bleiben.

Weitere Informationen finden sich auf der Homepage des Herstellers ([MLD03]).

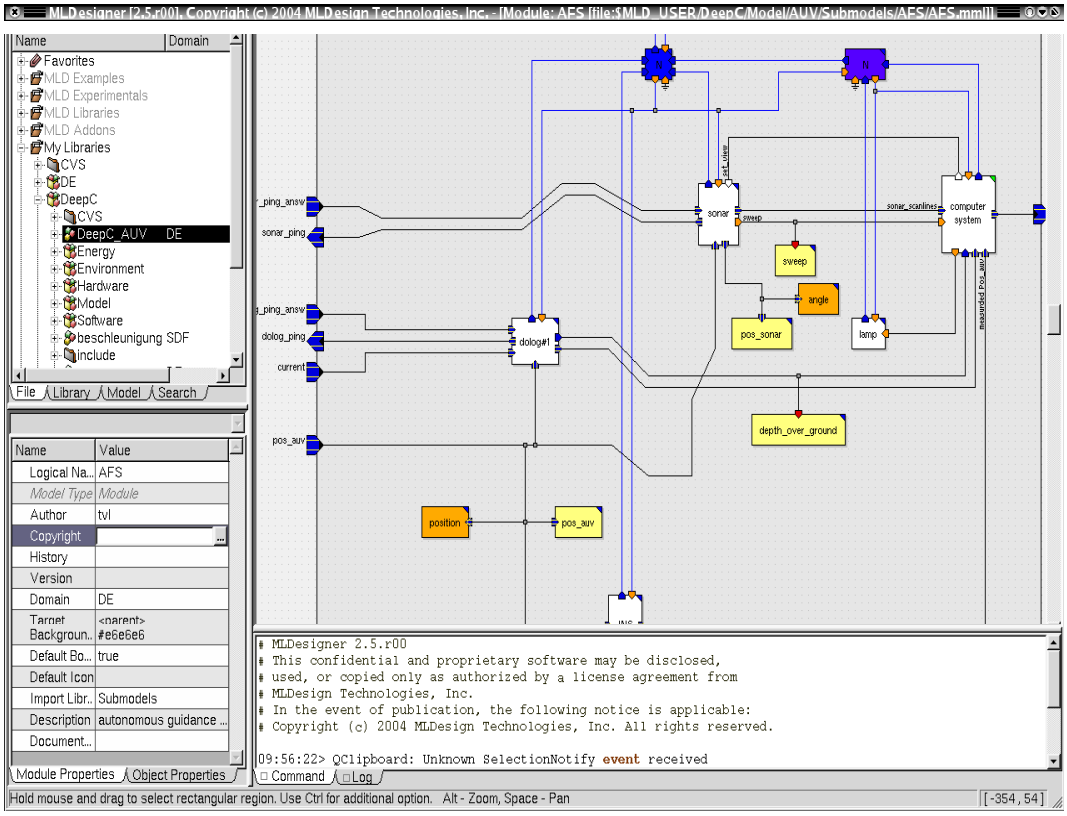


Abbildung 6.99: MLDesigner

6.4.2.2 Analyse MLDesigner

Da sich für mobile autonome Systeme Anpassungen am Mission Level Design vorgenommen werden mussten, wird im Folgenden untersucht, inwieweit sich auch bei der Simulation mit MLDesigner Anpassungen ergeben. Ausgangspunkt der vorgenommenen Analyse sind die vorgestellten Phasen der Simulation im Rahmen des Mission Level Designs, die von dem zur Simulationsdurchführung genutzten Programm unterstützt werden müssen.

6.4.2.2.1 Erstellung und Simulation von Modellen

Die Stärke von MLDesigner besteht in seinen sehr guten Fähigkeiten bei der Erstellung und Simulation von Modellen. Das Programm unterstützt die graphische Modellierung mittels einer übersichtlichen Benutzeroberfläche. Die Blöcke (Standardblöcke oder selbst erstelle) werden hierbei einfach in das Modell gezogen. Danach sind ihre Aus- und Eingänge zu verbinden und die Parameter zu setzen. Dabei steht für die Modellerstellung eine Vielzahl vorgefertigter Blöcke zur Verfügung, die zudem aus mehreren Domänen stammen und gemischt eingesetzt werden können. Damit ergibt sich die Möglichkeit einer hybriden

Simulation, die es dem Anwender erlaubt, für jedes Teilmodell den jeweils optimalen Modellierungsansatz zu nutzen.

6.4.2.2.2 Parametrisierung

Im Hinblick auf die Frage, inwieweit sich MLDesigner für die Missionshandhabung eignet, spielen die Möglichkeiten zur Modellparametrisierung eine entscheidende Rolle. Hierbei verfolgt MLDesigner den folgenden Ansatz: Die Parameter der einzelnen Blöcke lassen sich entweder direkt mit einem Wert belegen oder auf obere Modellebenen verlinken. Das Verlinken geschieht, indem auf der nächsthöheren Ebene ein Parameter vom selben Typ erzeugt wird und der Blockparameter an diesen Wert gekoppelt - verlinkt - wird. Mittels dieses Verfahrens lassen sich die Parameter Schritt für Schritt bis zur obersten Ebene verlinkt. Dort stehen sie dann für eine zentrale Belegung zur Verfügung.

Dieser von MLDesigner gewählte Ansatz für die Parametrisierung eignet sich lediglich für flache Modellhierarchien mit wenigen Parametern. Die Modellierung mobiler automatischer Systeme führt jedoch zu stark hierarchisch gegliederten Modellen. Somit müssen die Parameter über mehrere Ebenen hinweg verlinkt werden. Angesichts der großen Zahl von Verlinkungen werden dadurch Änderungen am Modell erschwert, da in diesem Fall alle entsprechenden Verlinkungen angepasst werden müssen. Außerdem birgt dies bei einer größeren Parameteranzahl schnell die Gefahr der Unübersichtlichkeit. Dies resultiert daraus, dass sich der Verwendungsort eines Parameters nur feststellen lässt, indem die Parameter der einzelnen Blöcke auf ihre verlinkten Parameter hin untersucht werden.

Die Eingabe von Missionen erlaubt MLDesigner nur durch direkte Vorgabe von Parameterwerten, wobei keine Unterscheidung der Parametertypen vorgesehen ist. Das bedeutet, dass der Nutzer bei der Parametrisierung für jede Mission selbst entscheiden muss, welche Werte wann zu ändern sind. Dabei müssen die Werte für jede Mission immer wieder von neuem eingeben werden, da MLDesigner keine Speicherung von Parametersets vorsieht. Es lässt sich zudem keine graphisch oder anderweitig unterstützte Eingabe realisieren, vielmehr ist die Eingabe eine rein textbasierte Wertvorgabe.

6.4.2.2.3 Simulationsauswertung

Für die missionsspezifische Auswertung der Simulationsergebnisse bietet MLDesigner standardmäßig eine Basisauswahl von Blöcken an: Es verfügt über Blöcke zur Kurvendarstellung (2D Plot) und zur Anzeige von Zahlenwerten. Des weiteren stehen über die Tcl/Tk-Anbindung diverse einfache Ausgabemöglichkeiten bereit. Zusätzlich lassen sich die Daten zur Weiterverarbeitung in Dateien schreiben. Damit stellt MLDesigner die grundlegenden Möglichkeiten für eine Auswertung von Simulationsergebnissen bereit.

Für den Einsatz im Rahmen des Mission Level Designs ergeben sich jedoch drei prinzipielle Probleme:

- **Vermischung:**
  Die Aufbereitung der Daten für die Darstellung erfolgt in MLDesigner durch zusätzliche Blöcke, die den Darstellungsblöcken vorgelagert sind.

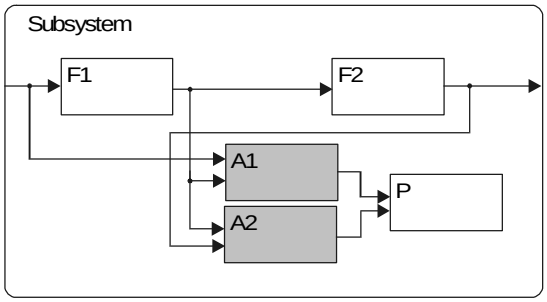


Abbildung 6.100: Beispiel mit Aufbereitung

Ein Beispiel gibt die Abbildung 6.100: Um die Ausgabe der Funktionsblöcke F1 und F2 mit P darstellen zu können, werden die der Aufbereitung dienenden Blöcke A1 und A1 zwischengeschaltet. Sie finden nur in der Darstellung Verwendung.
MLDesigner vermischt an dieser Stelle zwei verschiedene Aufgaben. Es integriert die Aufbereitung der Daten für die Simulationsauswertung in die Modellierung, was die Interpretierbarkeit des Modells negativ beeinflusst.

- **Fenster:**
  Ein weiteres Problem besteht in der Tatsache, dass MLDesigner für jeden der Ausgabeblöcke ein Fenster mit dem zugehörigen Plot erzeugt. Dabei lässt sich für jedes der Fenster dessen Position und die Größe einzeln durch Parameter des Ausgabeblocks einstellen. Bei einer großen Anzahl von Ausgabewerten sind jedoch viele Fenster offen, die sich teilweise überlagern, was eine gezielte Interpretation einzelner Verläufe erschwert.
- **Simulationsnähe:**
  Drittens erzwingt MLDesigner mit seiner simulationsnahen, sich direkt an die Simulation anschließenden Darstellung der Plots deren sofortige Auswertung. Zwar können Daten in Dateien geschrieben werden, aber das Programm ist nicht in der Lage, sie für eine spätere Auswertung direkt wieder anzuzeigen. Für eine große Anzahl von Missionen ist dies problematisch.
  Damit unterstützt MLDesigner zwar eine Speicherung der gewonnenen Ergebnisse, bietet aber keine die Möglichkeit mit den Ergebnisse außer direkt nach der Simulation zu arbeiten. Beispielsweise fällt ein direkter Vergleich zweier Simulationsläufe einer Mission schwer. Dies aber stellt eine wichtige Hilfe bei der Bewertung der am Design, und damit auch am Gesamtsystemmodell, vorgenommenen Veränderungen dar.

Die Analyse der Fähigkeiten von MLDesigner bezüglich der Anforderungen, die das Mission Level Design mobiler automatischer Systeme stellt, legt somit einige Probleme offen. Diese betreffen die Unterstützung von Missionen in der Parametrisierung und in der Auswertung, nicht jedoch die Modellierung und Simulation.

6.4.2.3 Framework zur Durchführung der Simulation

Um die Simulation durchführen zu können, wird im Folgendem ein neu entwickeltes Framework vorgestellt, das zwei wesentliche Prinzipien einführt, die aus der Analyse der Fähigkeiten von MLDesigner abgeleitet wurden:

- **Trennung von Modellierung und Missionshandhabung**
  Durch die Trennung von Modellierung und Missionshandhabung lässt sich die Unterstützung von Missionen ebenso wie eine erweiterte Funktionalität bei der Parametereingabe ermöglichen.

Die Unterstützung von Missionen impliziert, dass die Parametrisierung zwischen den verschiedenen Parametertypen unterscheidet. Auf diese Weise wird die Beschreibung der Missionen (mittels der Missionsparameter) von der Auslegung des Systems (mittels der Systemparameter) getrennt.
Die damit verbundene Vereinfachung und Unterstützung der Parametrisierung hat zudem zur Folge, dass auch Nutzer die Simulationen durchführen können, die nicht mit der Modellierung und mit den Modell vertraut sind.

- **Trennung von Modellnutzung und Auswertung**
  Die Trennung von Modellnutzung und Auswertung führt dazu, dass die Simulationsergebnisse zwischengespeichert werden müssen, was gleichzeitig ihre Archivierung erleichtert. Dies vereinfacht wiederum den Vergleich zwischen aktuellen und früheren Resultaten, weil das externe Programm immer auf gespeicherte Ergebnisse zugreift.
  Vorteilhaft ist des weiteren, dass sich sowohl die Aufbereitung, als auch die Darstellung getrennt von der Simulation erweitern lassen. Es ist also möglich, neue Aufbereitungsalgorithmen und Plots auch zu einem späteren Zeitpunkt in die Auswertung einfließen zu lassen, ohne Änderungen am Modell vornehmen oder neu simulieren zu müssen. Indem die Datenaufbereitung nach der Beendigung der Simulation durchgeführt wird, wird sie aus dem Modell entfernt, womit das Modell besser die Struktur des Systems abbildet.

Aus beiden Prinzipien ergibt sich eine Auftrennung der einzelnen Phasen zu verschiedenen Programmen. Dadurch können die benötigten Veränderungen im Bereich der Parametrisierung und der missionsspezifischen Auswertung erfolgen, ohne dass in das Programm MLDesigner eingegriffen wird.

Die Veränderungen werden mit Hilfe zweier neuer Programme realisiert, die im Zusammenspiel mit MLDesigner das Framework für das Mission Level Design mobiler automatischer Systeme bilden. In ihm übernehmen spezialisierte Programme die einzelnen Phasen, deren Zuordnung wie folgt lautet:

- **MLDesigner** wird für die Modellierung und die Simulation genutzt.
- Für die Handhabung der Missionen (also die Parametrisierung des Modells) wurde **MLEditor** neu entwickelt.
- **MLVisor**, ebenfalls neu entwickelt, ist speziell für die missionsspezifische Auswertung zuständig.

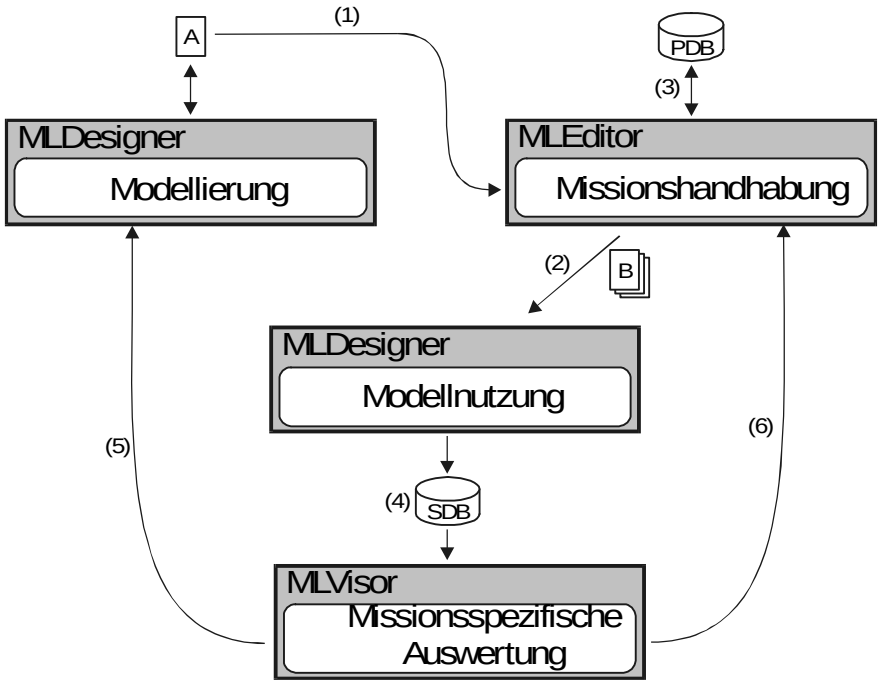


Abbildung 6.101: Konzept des Frameworks

Eine graphische Veranschaulichung des entwickelten Konzeptes für das Framework findet sich in Abbildung 6.101.

Die bloße Zuordnung der Programme des Frameworks zu den Aufgaben ist noch nicht ausreichend. Vielmehr ist die Betrachtung der Schnittstellen zwischen ihnen und damit des Aspekts des Datenaustausches notwendig. Im nachfolgenden Abschnitt wird dies geboten, indem auf die Schnittstellen eingegangen und die Zusammenarbeit der Programme beschrieben wird. Die einzelnen Punkte lassen sich dabei anhand der Abbildung 6.101 und der in ihr vergebenen Nummern nachvollziehen.

6.4.2.3.1 Parameterübergabe

Eine Erkenntnis des vorigen Abschnitts besteht in der Notwendigkeit, die Parametrisierung aus MLDesigner herauszulösen und diese Aufgabe an MLEditor zu übergeben, wodurch eine verbesserte Erstellung und Verwaltung der Missionen erreicht wird. Entsprechend müssen beiden Programme miteinander interagieren, da nur so die Missionen außerhalb des Modells verwaltet werden können.

Der gewählte Weg führt hierbei über die MLDesigner-Modelldateien. Dieses Vorgehen hat den Vorteil, dass die Änderungen für MLDesigner vollkommen transparent gestaltet werden können. Es erfordert jedoch einerseits, dass MLEditor die MLDesigner-Modelle lesen und die darin enthaltenen Parameter erfassen kann (1). Andererseits muss MLEditor in der Lage sein, die Modelle mit neuen Parametern für die Simulation zu schreiben (2). Auf der Basis der gelesenen Parameter wird somit von MLEditor die Parametrisierung des Modells unterstützt. Durch dieses Vorgehen des Frameworks entstehen zwei Typen von Modellen:

- **Entwurfsmodell (A):**
  Das Entwurfsmodell ist dasjenige MLDesigner-Modell, an dem der Entwurf vorgenommen wird, wobei es jedoch noch nicht parametrisiert ist.
- **Konfigurierte Simulationsmodelle (B):**
  Diese Simulationsmodelle werden von MLEditor automatisch aus dem Entwurfsmodell temporär erzeugt. Sie dienen der Simulation, denn sie stellen eine Version des

Entwurfsmodells dar, die für eine Mission parametrisiert wurde. Ihre Anzahl entspricht dabei der jeweiligen Anzahl von Missionen.

6.4.2.3.2 Parameterspeicherung

Die Parameter werden nun für verschiedene Modelle zentral in einer Datenbank verwaltet (3). Diese Parameterdatenbank (PDB) bietet folgenden Vorteil: Die Parametrisierung für eine Mission erfordert vom Nutzer nur die Angabe der Mission und nicht mehr das Einstellen der Werte, da die Parameterwerte gespeichert sind. Eine Mission spiegelt sich somit in einer Konfiguration des parametrisierten Modells wider. Folglich muss MLEditor für jede Mission eine parametrisierte Version des Entwurfsmodells erzeugen, welches dann mit MLDesigner zu simulieren ist. Die Missionssimulation erfolgt mithin durch die Teilschritte Modellierung (MLDesigner), Missionserstellung (MLEditor) und Simulation (MLDesigner).

6.4.2.3.3 Speicherung der Simulationsergebnisse

Durch die Trennung von Modell und Aufbereitung bzw. Darstellung wird es nötig, die Simulationsergebnisse zu speichern. Dafür wird im hier vorgestelltem Konzept eine Simulationsdatenbank (SDB) genutzt (4).

MLDesigner schreibt die Daten während der Simulation in die SDB, MLVisor nutzt sie für die Darstellung. Ein Vorteil der SDB liegt in der gleichzeitigen Archivierung der Ergebnisse. So lassen sich die Ergebnisse früherer Simulationen abrufen und direkt mit neueren Ergebnissen vergleichen. Außerdem kann die Auswertung der Missionen ohne Unterbrechung durch Simulationen und in beliebiger Reihenfolge durchgeführt werden. Da zusätzlich auf die Informationen aus der Parameterdatenbank zurückgegriffen werden kann, ist es zudem möglich, einen halbautomatischen Test auf die Einhaltung der System- oder der Entwurfsparameter durchzuführen.

Aus der Loslösung von direkten Benutzereingaben resultiert die Möglichkeit einer Zeitraffung der Simulation. Das heißt, die Simulation kann schneller als die Echtzeit ablaufen. Dies stellt bei einer großen Anzahl von Missionen und einer langen Missionsdauer einen Vorteil dar.

6.4.2.3.4 Nutzung der Ergebnisse

Die aus der Auswertung gewonnenen Erkenntnisse werden anschließend in das Entwurfsmodell übertragen. Dabei können Änderungen am Design des Systems vorgenommen werden (5). Eine andere Möglichkeit besteht in der Anpassung der Auslegung der Systemkomponenten über eine Änderung ihrer Systemparameter in MLEditor (6).

6.4.3 DeepC-Gesamtsystemmodell und Beispielmission

6.4.3.1 Gesamtsystemmodell

Ein weiterer wesentlicher Bestandteil dieser Arbeit - neben der Klärung der Frage wie die Simulation auf Missionsebene für DeepC durchgeführt werden kann - bestand in der Erstellung eines Gesamtsystemmodells für das autonome Unterwasserfahrzeug (AUV) DeepC. Diese Umsetzung der DeepC Spezifikation soll im Folgendem allgemein vorgestellt werden.

Abbildung 6.102 zeigt eine graphische Übersicht des mit MLDesigner erstellten Gesamtsystemmodells. In der graphischen Darstellung werden die einzelnen Komponenten durch Blöcke symbolisiert. Befindet sich ein Block innerhalb eines anderen, handelt es sich bei ihm um ein Subsystem. Damit gilt:

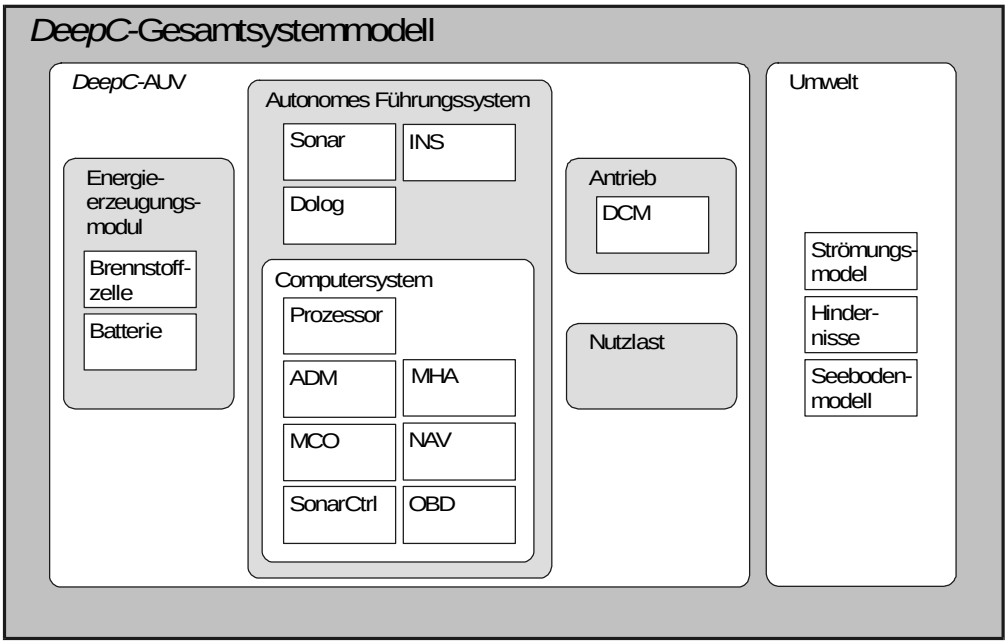


Abbildung 6.102: Gesamtsystemmodell (Übersicht)

- **(n)-Ebene:**
  Auf der obersten Ebene besteht das Gesamtsystemmodell aus dem AUV und seiner Umwelt.
- **(n-1)-Ebene:**
  Auf der (n-1)-Ebene untergliedert sich sowohl das Fahrzeug, als auch die Umwelt in Module.
  Für das Erstgenannte lassen sich vier Teilmodule unterscheiden: der Antrieb, das Autonome Führungssystem, das Nutzlast- sowie das Energieerzeugungsmodul. Sie sind Modelle der entsprechenden Systembestandteile: Das Energieerzeugungsmodul fasst die Systemteile Brennstoffzelle und Energieerzeugungseinrichtungen zusammen, während die Manövriereinrichtungen und das Antriebssystem den Antrieb bilden. Zunächst nicht modelliert wurden das Kühlsystem, das Notfallsystem und das Trimmsystem. Der Systembestandteil Struktur wird indirekt an verschiedenen Stellen im Modell abgebildet.
- Die Umwelt umfasst drei Modelle, nämlich die Hindernisse, ein Strömungsmodell und ein Modell des Seebodens.
- **(n-2)-Ebene:**
  Jedes der Fahrzeugteilmodule auf der Ebene (n-1) besteht seinerseits aus Subkomponenten.
  Das Energieerzeugungsmodul beinhaltet die Brennstoffzelle und die Pufferbatterien. Für den Antrieb existiert lediglich eine DCM (Dynamic Controlled Motion) genannte Komponente, die ein Modell des geregelten Fahrzeugverhaltens beinhaltet. In diese sind die Funktionen der Manövriereinrichtungen integriert, so dass hierfür kein separates Teilmodul modelliert werden musste.
  Das autonome Führungssystem besteht aus der Sensorik mit Sonar, INS (Inertial Navigation System) und Dolog sowie dem Rechnersystem, auf welchem die Steuersoftware läuft.
- **(n-3)-Ebene:**
  Auf dieser Ebene untergliedert sich das Computersystem in seine Hardware (Prozessor) und die einzelnen Softwaremodule, die auf der Hardware laufen (ADM, MHA, MCO, NAV, OBD und SonarCtrl).

Damit eine strukturierte Erklärung des Gesamtverhaltens erreicht werden kann, soll das Gesamtsystemmodell nun aus einzelnen Sichten vorgestellt werden, die jeweils einen

Modellaspekt hervorheben. Dabei werden die zugehörigen Komponenten in ihrer Zusammenarbeit beschrieben. Die Sichten überlagern sich gegenseitig und sind insofern nicht unabhängig voneinander.

Die Modellierung des Gesamtsystemmodells erforderte dabei die Erstellung zahlreicher neuer Module für die einzelnen Komponenten des AUVs. Für eine detaillierte Beschreibung dieser Komponentenmodelle muss an dieser Stelle auf die Dokumentation des Gesamtsystemmodells verwiesen werden.

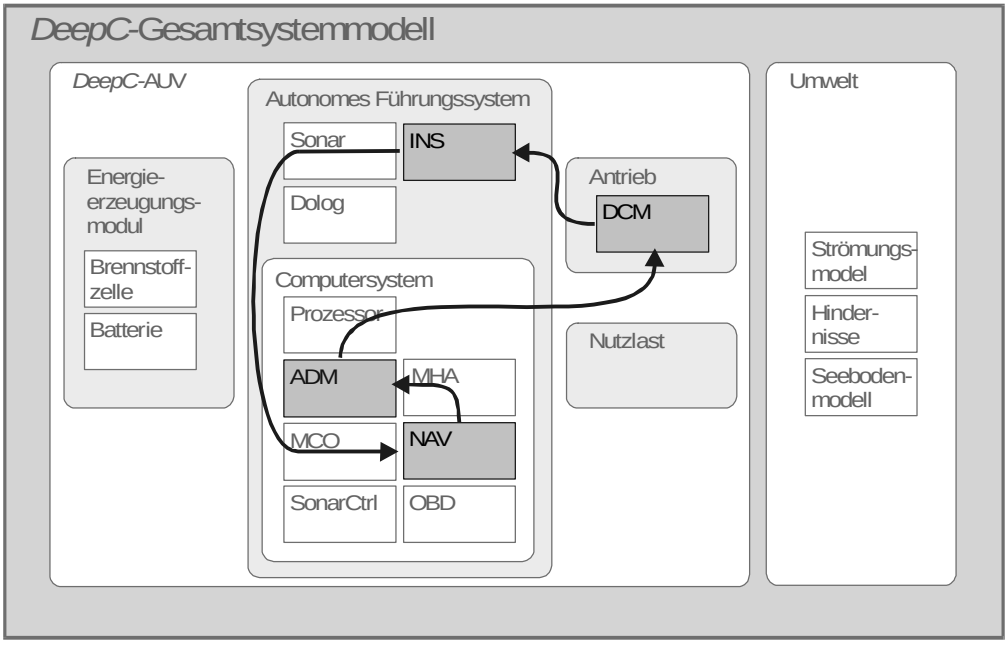


Abbildung 6.103: Teilsicht: Antrieb

6.4.3.1.1 Bewegung

Ein zentraler Bereich für die Modellierung eines AUV ist die Nachbildung seiner Bewegung im Raum. Das Fahrzeug soll sich im dreidimensionalen Raum mit allen sechs Freiheitsgraden bewegen können und dabei die charakteristische Dynamik aufweisen.

Abbildung 6.103 stellt die hierbei beteiligten Blöcke sowie die Verbindungen zwischen ihnen dar. Sie nutzt dazu als Basis die bereits vorgestellte graphische Darstellung des Gesamtsystemmodells. In dieser werden nun diejenigen Bestandteile hervorgehoben, die in Verbindung mit der Bewegung stehen. Hierbei handelt es sich um die Blöcke DCM (geregeltes Fahrzeugverhalten), INS (inertiales Navigationssystem), NAV (Navigation) und ADM (Basisregler). Die folgende Auflistung zeigt deren schrittweises Zusammenspiel auf:

1. Die Dynamik wird von einem Modell des geregelten Fahrzeugverhaltens erzeugt. Es erhält die Steuersignale Kurs, Geschwindigkeit und Tiefe und erzeugt daraus die aktuelle Fahrzeugposition.
2. Das Fahrzeug nutzt für seine Positionsbestimmung ein inertiales Navigationssystem. Im Modell werden die vom Fahrzeug kommenden, exakten Positionsdaten zum INS geleitet. Dieses generiert aus ihnen die gemessene AUV Position.
3. Diese Position wird vom INS an das Softwaremodul Navigation übermittelt, das die Daten für andere Softwaremodule zugänglich macht.
4. Auf Basis der aktuell gemessenen Position erzeugt der als Softwaremodul realisierte Basisregler ADM die Steuersignale. Er hält das Fahrzeug auf einem Basisbahnelement, bei dem es sich um eine Linie oder ein Kreissegment handeln kann. ADM gibt die Steuersignale für DCM vor (zurück zu 1.).

6.4.3.1.2 Umwelt und Sensorik

Die Modellierung der Umwelt bildet einen wesentlichen Bestandteil des Gesamtsystemmodells. Gleichzeitig hierzu muss jedoch auch die Sensorik zum Wahrnehmen der Umwelt nachgebildet werden.

Abbildung 6.104 hebt die an diesem Vorgang beteiligten Blöcke hervor.

- Hierbei sind erstens die Hindernisse, das Bodenmodell, das Sonar und das Dolog zu nennen. Von ihnen sind die Hindernisse und das Bodenmodell zur Umwelt zu zählen, während das Sonar und das Dolog der Sensorik zuzurechnen sind.
  Die Zusammenarbeit zwischen der Sensorik und der Umwelt geschieht über Anfragen (Pings). Diese bilden jeweils einen Suchstrahl nach und werden sowohl vom Sonar, als auch vom Dolog an die Umwelt gesendet. Dort werden sie entgegengenommen und an das Seebodenmodell und das Modell der Hindernisse weitergeleitet. Die Antworten werden ausgewertet und an dasjenige Sensormodul weitergereicht, welches die Anfrage gestellt hat. Durch dieses Vorgehen wird es möglich, auf einfache Art und Weise weitere Umwelteinflüsse hinzuzufügen.
- Ein zweiter Umwelteinfluss wird durch das Strömungsmodell modelliert. Es erzeugt für jeden Punkt im Raum einen Geschwindigkeitsvektor. Dieser vom Dolog gemessene Vektor wird bei der Generierung der Position in DCM berücksichtigt und beeinflusst so die Bewegung des Fahrzeugs im Raum.

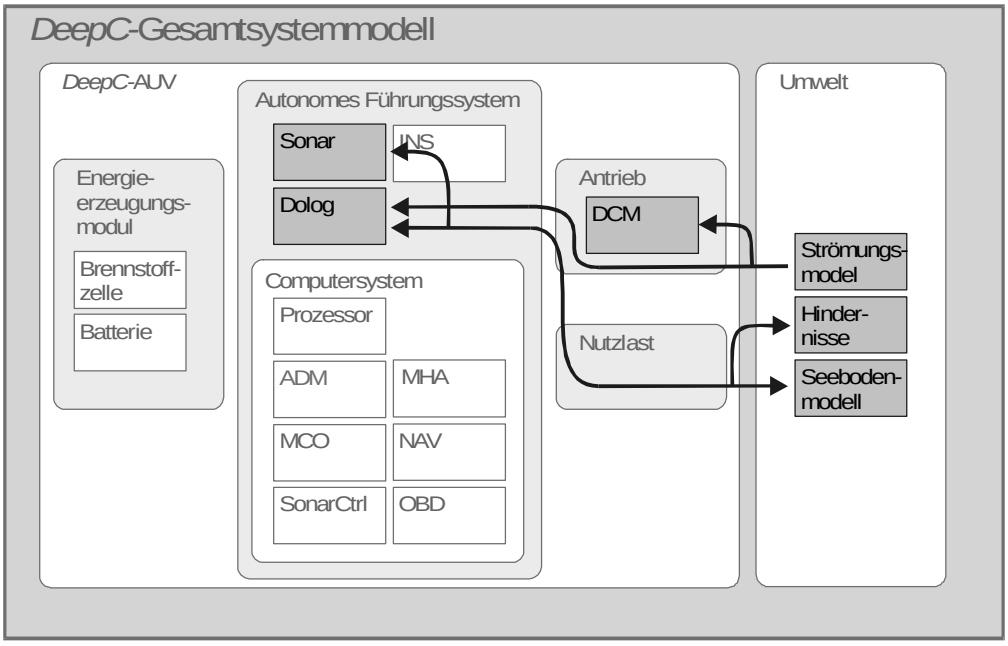


Abbildung 6.104: Teilsicht: Umwelt und Sensorik

6.4.3.1.3 Computersystem

Das Computersystem ist ein wesentlicher Bestandteil des Systems, da auf ihm die Softwareanwendungen zur Erzeugung der maschinellen Intelligenz laufen. Aus diesem Grund wurde die Softwarestruktur des AUV DeepC nachmodelliert. Sie besteht aus verschiedenen Softwaremodulen, die jeweils spezifische Aufgaben erfüllen. Abbildung 6.105 hebt nicht nur die Softwaremodule, sondern auch die dazugehörige Hardware hervor. Letztere modelliert die wichtige Ressource Rechenleistung.

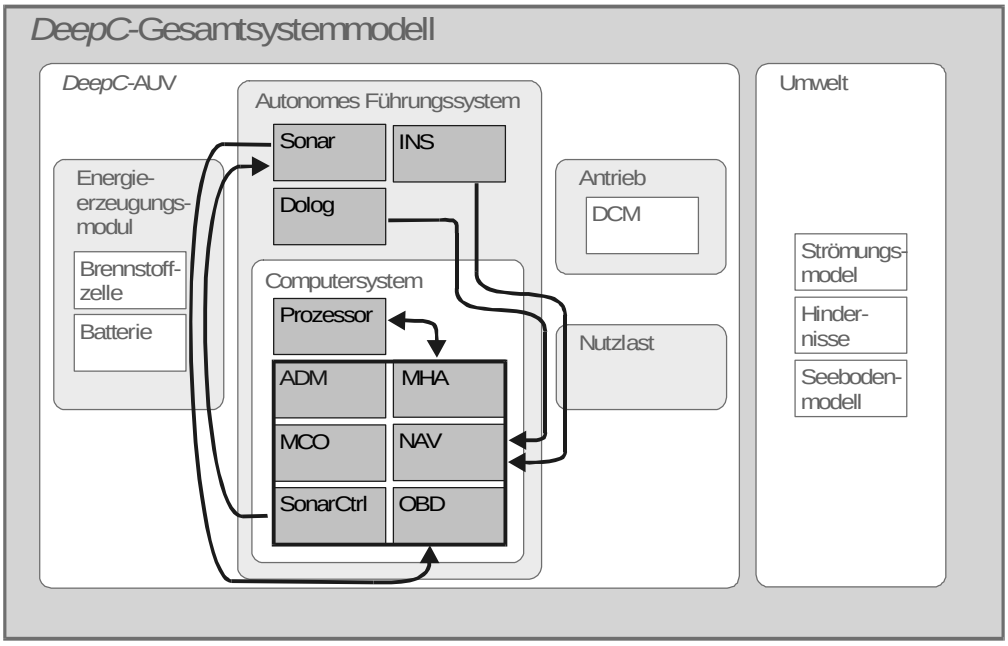


Abbildung 6.105: Teilsicht: Computer

In diesem Kontext lassen sich zwei unterschiedliche Teilsichten unterscheiden:

- Erstens arbeiten Hardware und Software nicht unabhängig voneinander, vielmehr nutzt die Software die vorhandene Hardware. In der Simulation wird dieser Umstand dadurch modelliert, dass die Softwaremodule eine gewisse Rechenarbeit benötigen. Die Ablaufzeit zur Erfüllung dieser Arbeit ist dabei nicht festgeschrieben, sondern von der Hardware und ihrer Auslastung abhängig. Aus diesem Grund besitzt jeder Rechner eine parametrisierbare Menge an Rechenleistung, die prioritätsgesteuert auf die parallelen Anfragen der Softwaremodule verteilt wird. Ist die Arbeit erledigt, werden die Softwaremodule benachrichtigt.
- Die zweite Teilsicht betrifft die Zusammenarbeit der Softwaremodule untereinander. Das AUV DeepC besitzt eine hierarchische Softwarestruktur, wie sie Abbildung 6.106 zeigt. Alle farblich hinterlegten Blöcke sind im Gesamtsystemmodell berücksichtigt.

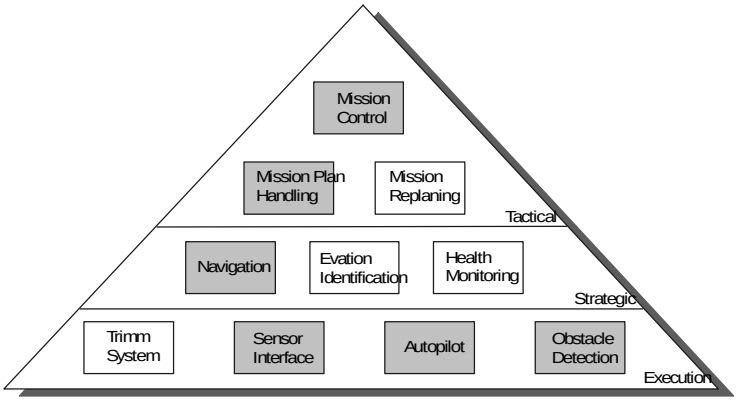


Abbildung 6.106: Vollständige DeepC Softwarestruktur

An der Spitze steht die taktische Ebene. Sie wird vom Mission Controller (MCO), Mission Plan Handling (MHA) und dem Mission Replaning gebildet. Der Mission Controller stellt die höchste Entscheidungsinstanz dar und kann insofern als der Kapitän des Fahrzeugs bezeichnet werden. Das Mission Plan Handling verwaltet die einzelnen Elemente des Missionsplanes, welcher den Ablauf jeder Mission steuert,

während das Mission Replaning für eine ereignisgesteuerte Umplanung des Missionsplanes verantwortlich ist.
Die strategische Ebene wird von Softwaremodulen gebildet, die spezielle strategische Aufgaben übernehmen. Navigation (NAV) übernimmt alle im Zusammenhang mit der Navigation nötigen Aufgaben. Das Health Monitoring überwacht das Funktionieren wichtiger Systembestandteile und EvasionAndIdentification steuert das Ausweichen beziehungsweise das Identifizieren von Objekten.
Auf der untersten, der ausführenden Ebene finden sich Softwaremodule, die direkt mit der Sensorik beziehungsweise Aktorik interagieren. Die Hinderniserkennung (Object Detection, OBD) wertet als Basis der Kollisionsvermeidung die Sonarbilder aus. Der Autopilot (ADM) erzeugt die Steuersignale für den Antrieb, das Sensor Interface übernimmt die Kommunikation mit der Sensorik, während Trimm System für die Trimmung des Fahrzeugs sorgt.

6.4.3.1.4 Energie

Der Energiekreislauf und die daran beteiligten Komponenten werden in Abbildung 6.107 hervorgehoben. Gespeist wird der Energiekreislauf von den Brennstoffzellen. Diese sind über Pufferbatterien mit den zahlreichen Abnehmern verknüpft, wobei diese Verknüpfung einem Energienetz entspricht.

Als Verbraucher treten unter anderem der Antrieb (DCM), die Prozessoren sowie die gesamte Sensorik (Sonar, Dolog, INS) auf. Einzelne Komponenten sind zuschaltbar oder ihr Energieverbrauch wird lastabhängig modelliert. Dies trifft beispielsweise für die Lampen (zuschaltbar) sowie für den Antrieb zu, welcher in Abhängigkeit von der Geschwindigkeit mehr oder weniger Energie verbraucht (lastabhängig).

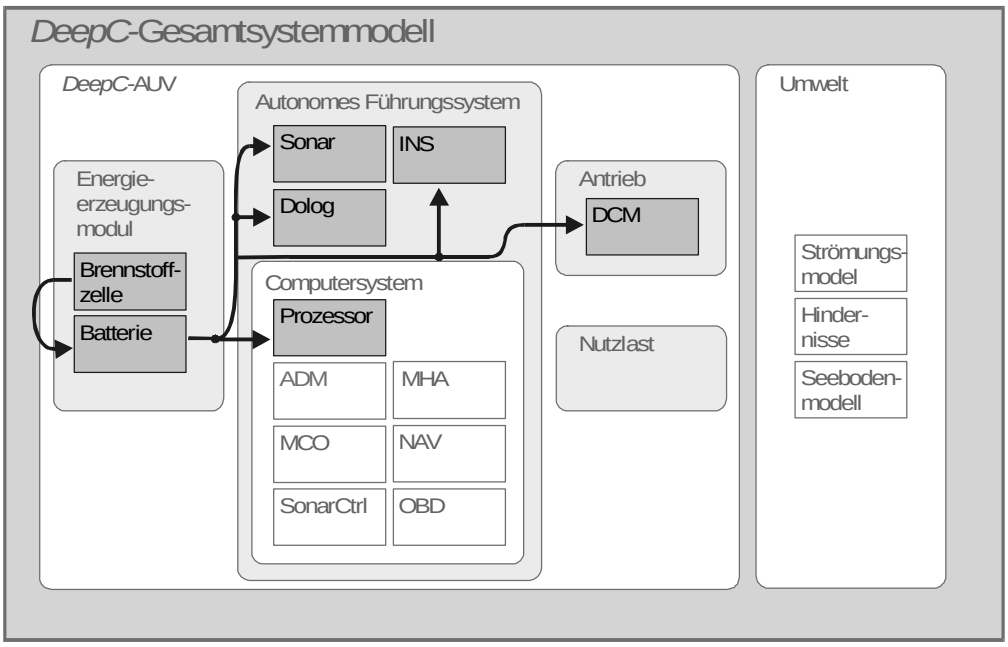


Abbildung 6.107: Teilsicht: Energie

6.4.3.2 Beispielmission

Wie der einführende Abschnitt zum Mission Level Design gezeigt habt, sind die Missionen neben dem Gesamtsystemmodell ein elementarer Bestandteil des Konzeptes. Im Folgenden soll deshalb ihre Beschreibung anhand eines exemplarischen Nutzungsszenarios erläutert werden. Die Beschreibung erfolgt mittels der Missionsparameter des Gesamtsystemmodells.

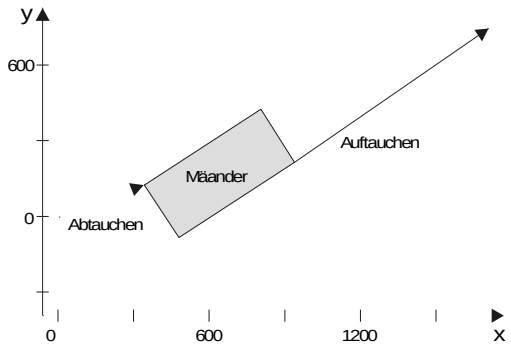


Abbildung 6.108: Manöver der Beispielmission

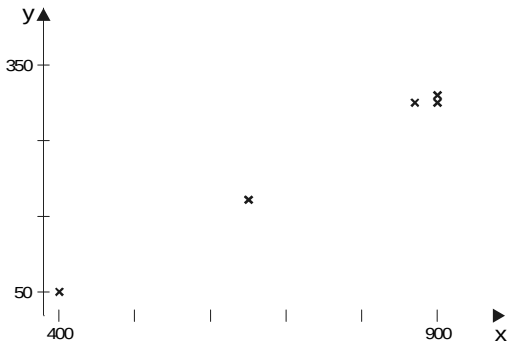


Abbildung 6.109: Hindernisse der Beispielmission

Diese Beispielmission besteht aus einer dreistufige Fahrt: Zunächst taucht das Fahrzeug in eine Tiefe von 100 m ab, absolviert dort einen Mäander und taucht anschließend wieder auf (siehe Abbildung 6.108) In der Nähe der geplanten Fahrstrecke befinden sich in der Tiefe des Mäanders fünf Hindernisse. Abbildung 6.109 zeigt ihre Lage aus der Draufsicht. Das Ziel der Mission besteht darin, alle diese Hindernisse zu erkennen. Der Ausgangszustand des Fahrzeugs ist dadurch charakterisiert, dass es seine Fahrt vollaufgetankt und funktionstüchtig am Startpunkt der Mission aufnimmt.

Diese Mission wird mit den Missionsparametern nachgebildet. Den ersten Teil die Fahrtroute beschreibt der Missionsplan. Bei ihm handelt es sich um ein Protokoll, das die zu absolvierenden Manöver festlegt. Das Protokoll wird auch von der Simulation unterstützt, wobei der Missionsplan vom Parameter *MissionControl.MissionPlan* aufgenommen wird.

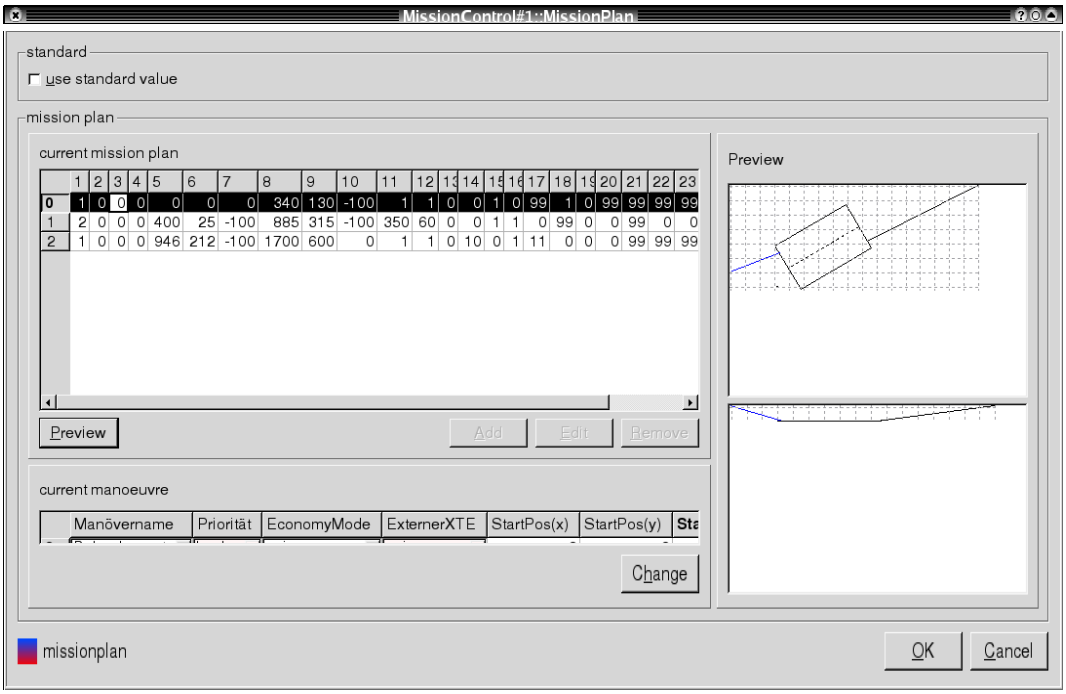


Abbildung 6.110: MLEditor Missionsplan Plugin

Um die Eingabe von Missionsplänen zu vereinfachen, existiert für diese ein spezielles MLEditor-Plugin (siehe Abbildung 6.110). Es gliedert sich in drei Bereiche: Links oben wird der Missionsplan in seiner Simulationsstruktur angezeigt (Matrix mit codierten Informationen). Unterhalb der Matrix befindet sich ein Bereich, in dem je ein Manöver bearbeitet werden kann. An dieser Stelle wird das offizielle Protokoll anzeigt, das heißt, die Informationen werden in verbaler Form beschrieben. Die rechte Seite bietet eine Vorschau über den Fahrtverlauf des Missionsplans (Projektionen auf die x-y und die x-z-Ebene). Durch das Plugin vereinfacht sich die Handhabung wesentlich. Die einzelnen Manöver können

bequem eingegeben werden, wobei sämtliche Werte entsprechend des Protokolls übertitelt werden. Über die Vorschaufunktion ist jederzeit der aktuelle Bearbeitungsstand ersichtlich.

Tabelle 1.1 bietet eine vollständige Übersicht über die Werte, die die Missionsparameter für die Beispielmission annehmen. *objects.obstacles* nimmt die Objektpositionen der Hindernisse auf. Die Testmission erhält die Nummer Null (*PrinterDBTop.mission*), alle *EnNode.Broken*-Parameter werden auf false gesetzt (Standard). Da das Fahrzeug voll aufgeladen ist, verzeichnen *Battery.Level* und *fuelcell.Level* den Wert 100. Die Startposition wird in den lokalen Nullpunkt gesetzt (*DCM.{x0, y0, z0, phi0, theta0, psi0 }*), an dem das Fahrzeug aus dem Stillstand startet (*DCM.u0*). Global befindet sich das Fahrzeug bei (*bottom.start_lat_lon*). Die Standardverzeichnisse mit den Boden- und Strömungsdaten bleiben unverändert (*bottom.filedir, current.path*).

Tabelle 6.9: Beispielmission für DeepC

Parameter	Wert
MissionControl.MissionPlan	<Matrix>
MissionControl.StartupModules	(10, 1), (1, 1), (2, 1), (3, 1), (4, 2), (11, 2)
PrinterDBTop.mission	0
EnNode.Broken (mehrere)	false
Battery.Level	100
fuelcell.Level	100
DCM.{x0, y0, z0, phi0, theta0, psi0 }	0
DCM.u0	0
objects.obstacles	(400, 50, -100), (650, 172, -100), (870, 300, -100), (900, 300, -100), (900, 310, -100)
bottom.filedir	$MLD_USER/DeepC/include/netCDF/data/
bottom.start_lat_lon	(2.1, 3.1)
current.path	$MLD_USER/DeepC/include/current/

6.4.3.3 Auswertung der Beispielmission

Nachdem die Parametrisierung des MLDesigner DeepC Gesamtsystemmodells vorgestellt wurde, wird jetzt anhand der Beispielmission präsentiert, wie die missionsspezifische Auswertung mit MLVisor erfolgt und welche Anzeigen sich aus den Ergebnisdaten generieren lassen. Um die Ergebnisdaten zu erhalten, wurden das konfigurierte Modell der Mission zuvor mit MLDesigner simuliert, wobei die entstehenden Ergebnisse in die Simulationsdatenbank geschrieben wurden.

Abbildung 6.111 zeigt MLVisor mit einer seiner Auswertungsansichten, die aus vier Anzeigen (Views), die auf Simulationsergebnissen basieren, besteht. Da sich diese Ansichten speichern und wieder aufrufen lassen, kann für jede Mission eine speziell angepasste Auswertungsoberfläche mit den benötigten Darstellungen zusammengestellt werden, die wiederverwendbar ist. Hierdurch wird der Umgang mit den unveränderlichen Missionen bei einem veränderlichen Gesamtsystemmodell erheblich erleichtert.

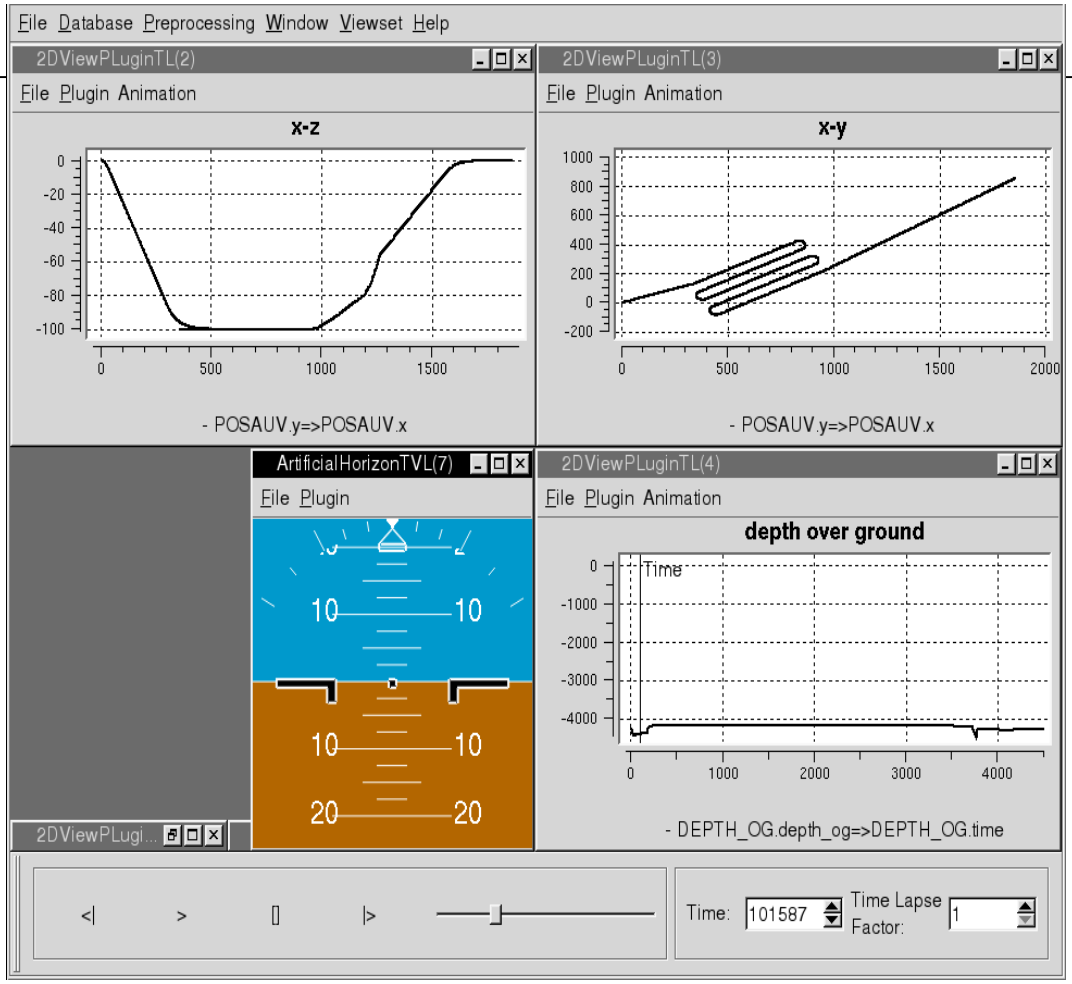


Abbildung 6.111: MLVisor-Hauptfenster

Die Simulationsergebnisse der Beispielmission werden nachfolgend anhand der Sichten des Modells präsentiert.

6.4.3.3.1 Bewegung

Die Bewegung der Fahrzeugs vollzieht sich, indem der vorgegebene Missionsplan abgefahren wird. Abbildung 6.112 zeigt die sich dabei ergebende Trajektorie des AUV aus der Draufsicht (x-y-Ebene). Deutlich ist das Abfahren des Missionsplanes - besonders das des Mäanders - nachvollziehbar. Aus dieser Sicht lassen sich die Ab- und Auftauchphase jedoch schwierig erkennen.

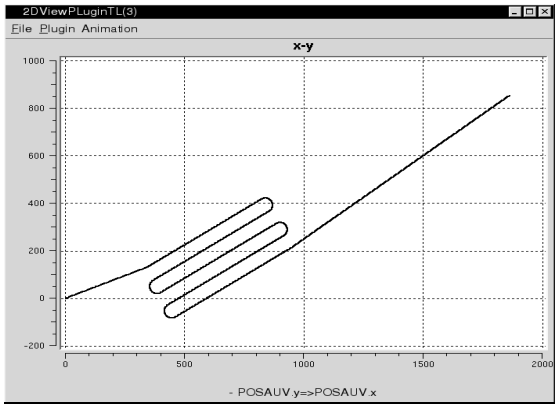


Abbildung 6.112: Trajektorie (x-y-Ebene)

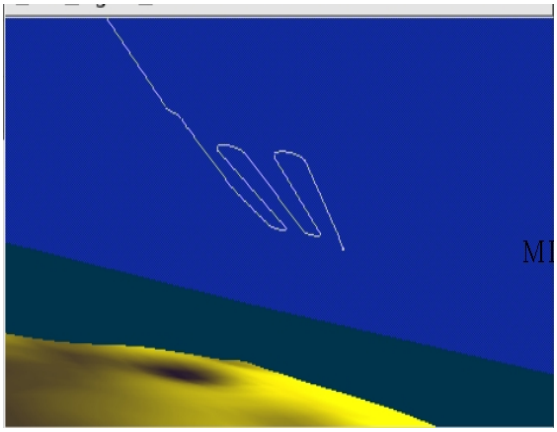


Abbildung 6.113: Trajektorie (3D)

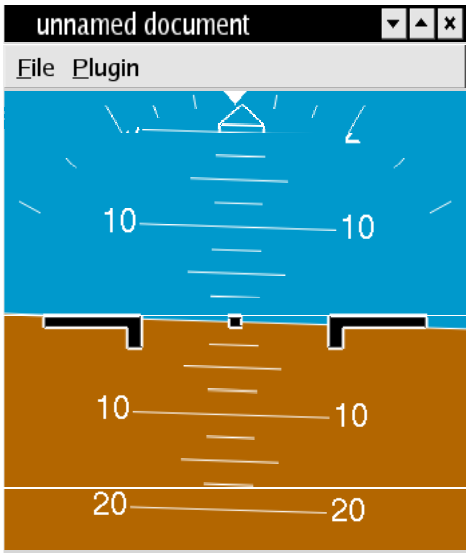


Abbildung 6.114: Künstlicher Horizont

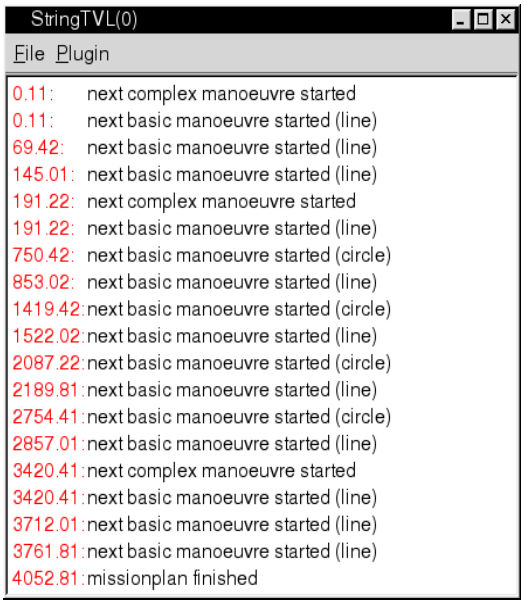


Abbildung 6.115: Meldungen

Durch ihre Projektion auf die Achsen geben die bisherigen Darstellungen die tatsächliche Bewegung im Raum jedoch nur eingeschränkt wieder. Aus diesem Grund steht für die Auswertung zusätzlich eine dreidimensionale Darstellung zur Verfügung (Abbildung 6.113). Sie ermöglicht die Betrachtung der Trajektorie durch Drehung im Raum sowie im animierten zeitlichen Ablauf. Gleichzeitig lassen sich zusätzliche Objekte, wie der Seeboden oder die Wasseroberfläche, in den dargestellten Raum integrieren.

Eine Möglichkeit, eine Vorstellung von der Bewegung des Fahrzeugs aus dessen Sicht zu erlangen, bietet die Darstellung seiner Nick- und Roll-Lage mittels eines künstlichen Horizonts. Abbildung 6.114 zeigt dieses aus der Luftfahrt bekannte Instrument.

Ein anderer Aspekt, der sich anhand der bisher vorgestellten Ansichten schwer untersuchen lässt, sind die Zeitpunkte, an denen die einzelnen Manöver des Missionsplanes eingeleitet werden. Das Softwaremodul MissionControl logt die entsprechenden internen Meldungen mit und stellt diese für die Auswertung bereit. Diese Meldungen lassen sich mit der Zeit ihres Auftretens darstellen, was Abbildung 6.115 zeigt. Die Dauer der Mission ist so beispielsweise anhand der letzten Meldung (MHA FINISHED) exakt ablesbar. Für die Beispielmission sind dies 4053 Sekunden (1 Stunde, 7 Minuten und 33 Sekunden).

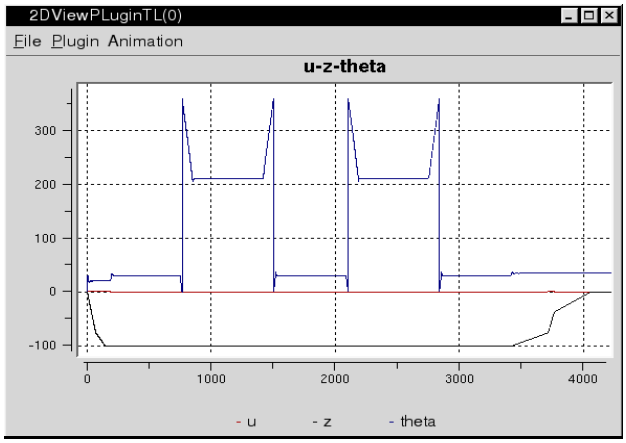


Abbildung 6.116: Steuerkommandos

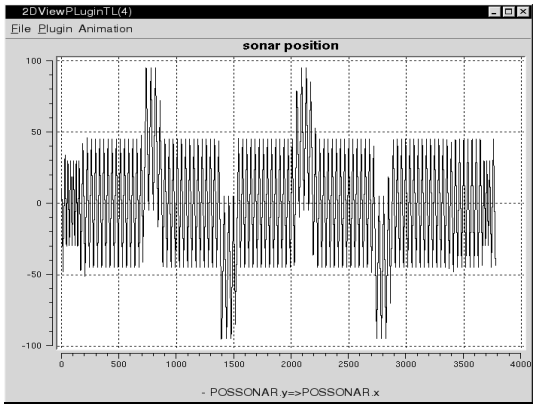


Abbildung 6.117: Steuerkommandos

Abbildung 6.116 zeigt die Steuerkommandos des DynamicManoeuringSystems, die die gefahrene Trajektorie erzeugen. Durch sie lassen sich die Steuervorgaben des Reglers nachvollziehen. Deutlich sind die Kurs- und die Tiefenvorgabe zu erkennen. Das Kurskommando schwankt zwischen den Werten 0 und 360°. Die waagerechten Linien repräsentieren einen geraden Kursverlauf, während die Schrägen den Kurven des Mäanders mit ihrer konstanten Kursänderung entsprechen. Die senkrechten Sprünge werden dabei vom charakteristischen 360°- Übergang hervorgerufen. In der Tiefenvorgabe sind nochmals das Ab- und das Auftauchen erkennbar. Das Geschwindigkeitskommando ist aufgrund seines vergleichsweise geringen Wertebereiches in dieser Ansicht schwer zu unterscheiden.

6.4.3.3.2 Umwelt und Sensorik

Die Ansteuerung des Sonars durch SonarCtrl lässt sich aus Abbildung 6.117 entnehmen. Sie zeigt in Sonarkoordinaten die Ausrichtung des Sonars über die Zeit. In der Ansicht sind deutlich die einzelnen Linien erkennbar, die sich durch das Schwenken des Sonars ergeben. Die vertikale Verschiebung der Linien über die Zeit verdeutlicht, wie das Softwaremodul den Sichtbereich im Missionsverlauf (genauer: vor und während der Kurven) anpasst und so dafür sorgt, dass das AUV die Kurven vor der Fahrt scannt.

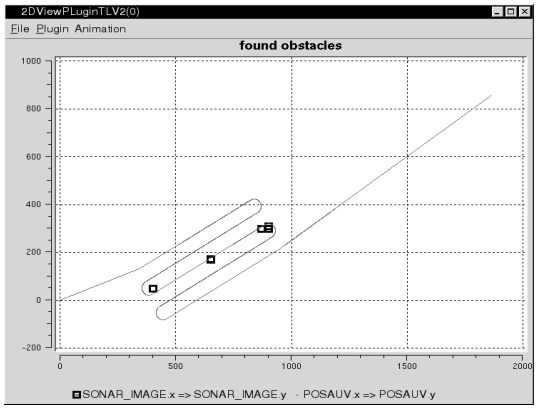


Abbildung 6.118: Erkannte Hindernisse

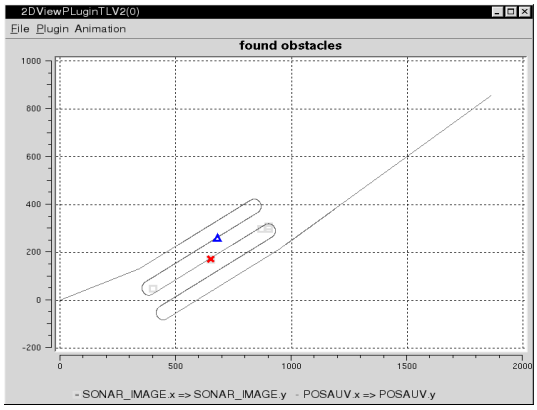


Abbildung 6.119: Erkannte Hindernisse (t=1061s)

Abbildung 6.118 zeigt zusammen mit der Trajektorie die von der Software erkannten Hindernisse (□). Der Vergleich mit Abbildung 6.119 zeigt, dass alle vorhandenen Hindernisse erkannt wurden. Durch die Einbeziehung dieser Daten in die animierte Darstellung der Trajektorie lässt sich zusätzlich der Zeitpunkt bestimmen, an dem das Fahrzeug die Objekte registriert hat (vgl. Abbildung 6.119). Die Position des AUV wird dabei durch das Symbol ∇ dargestellt, während die Hindernispositionen mit □ (unerkannt) und * (erkannt) markiert werden.

6.4.3.3.3 Computer

In der Auslastung der Prozessoren zeigt sich, wie gut die Ressource Prozessorarbeit dimensioniert ist. Abbildung 1.24 stellt die Ergebnisse der Simulation für beide Prozessoren dar. Computer 1 verzeichnet eine relativ konstante Grundlast, während Computer 2 eine sporadische Nutzung aufweist.

6.4.3.3.4 Energie

Neben der für eine Mission benötigten Energiemenge ist vor allem der Verlauf des Energieverbrauches von Interesse. Abbildung 1.25 bietet eine entsprechende Darstellung für die Brennstoffzelle. Deren Füllstand nimmt im Missionsverlauf kontinuierlich von 100 Prozent auf einen Wert von 99,4 Prozent ab.

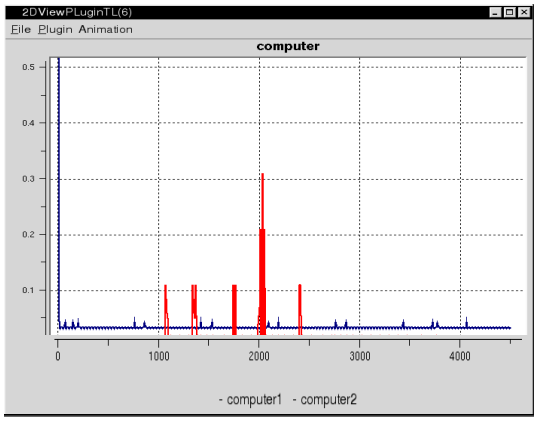


Abbildung 6.120: Prozessorauslastung

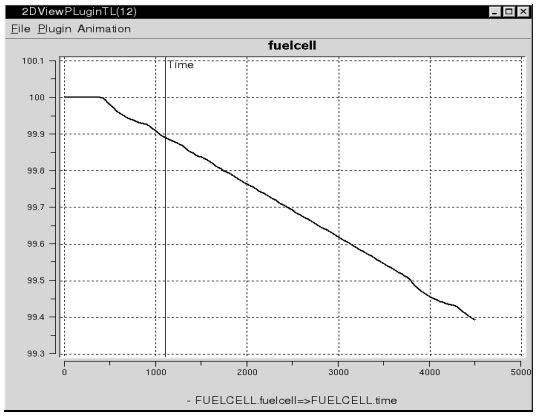


Abbildung 6.121: Füllstand der Brennstoffzelle

6.4.3.3.5 Fazit

Die aus der Beispielmission abgeleiteten Ergebnisse lassen sich wie folgt zusammenfassen:

1. Der vorgegebene Missionsplan wurde korrekt abgefahren (Abtauchen, Mäander, Auftauchen).
2. Alle platzierten Objekte wurden vom Sonar und der zugehörigen Auswertung erkannt.
3. Die Computer sind wenig ausgelastet und verfügen in dieser Mission über große Reserven.
4. Die Mission verbraucht angesichts ihrer kurzen Dauer nur einen geringen Teil der verfügbaren Energie.

Wie diese Ergebnisse zeigen, konnte das Fahrzeug die durch die Beispielmission gestellten Anforderungen voll erfüllen.

6.4.4 Zusammenfassung

Gegenstand dieser Arbeit war das Mission Level Design für das AUV DeepC. Im Mittelpunkt steht dabei die Simulation des Fahrzeugs auf Missionsebene. Sie verfolgt das Ziel, möglichst frühzeitig im Entwurfsprozess Aussagen bezüglich der Leistungsfähigkeit des Systems zu gewinnen und mögliche Probleme zu erkennen.

Beim Mission Level Design handelt es sich um eine missionsbezogene modellgestützte Systementwurfsmethode für komplexe Systeme. Sie überführt die Systemspezifikation in ein ausführbares Gesamtsystemmodell auf Systemebene und simuliert dieses. Gegenüber anderen Entwurfsmethoden zeichnet sich das Mission Level Design durch die Einbeziehung der Missionen aus. Bei ihnen handelt es sich um typische Einsatzszenarien des zu entwerfenden Systems. Durch die missionsbezogene Simulation wird sichergestellt, dass das System die angestrebten Eigenschaften erreicht.

Die vorliegende Arbeit identifizierte die Anforderungen, die die Modellierung mobiler automatischer Systeme wie DeepC an das Mission Level Design stellt. Dabei stellt sie fest, dass der Entwurf dieser Systeme eine Reihe von Erweiterungen des Modellierungsprozesses notwendig macht: Zum einen muss das Gesamtsystemmodell in die Teilmodelle Systemmodell und Umwelt untergliedert werden. Zum anderen erfordert die Einbeziehung der Missionen eine Erweiterung des Konzepts der Simulationsdurchführung um die neue Phase der Missionshandhabung und die Auswertung muss missionsspezifischen erfolgen.

Auf Basis dieser theoretischer Überlegungen entwickelt die Arbeit ein Framework für das Mission Level Design, das die praktische Durchführung von Simulationen auf Missionsebene ermöglicht. Sein Hauptbestandteil ist das kommerzielle Entwurfsprogramm MLDesigner, das um die neuentwickelten Programme MLEditor und MLVisor erweitert wird.

Mit Hilfe dieses Frameworks wurde für das DeepC -Projekt ein Gesamtsystemmodell des AUVs DeepC erstellt, um damit Untersuchungen auf Missionsebene durchzuführen. Es besteht aus einer virtuellen Umwelt und einem Fahrzeugmodell, welches neben der Dynamik, der Sensorik und Aktorik auch die Softwarestruktur und ihre Abarbeitung auf einer virtuellen Hardware nachbildet. Des weiteren integriert das Modell den energetischen Aspekt, der als begrenzte Ressource für ein autonomes Unterwasserfahrzeug von besonderer Bedeutung ist. Anhand dieses praktischen Beispiels wird die Durchführung von Simulationen auf Missionsebene demonstriert.

6.4.5 Literatur

[AL03] Andrejevi'c, Miona ; Liebezeit, Thomas: *Modeling of the energy system for the mission level design of the DeepC AUV*. In: Proceedings of the twelfth international scientific and applied science conference ELECTRONICS, ET'2003, Sozopol, Bulgaria, 2003

[Lie02] Liebezeit, Thomas: *Mission Level Design of Autonomous Underwater Vehicles*. In: Nahavandi, Saeid (Hrsg.): Proceedings of the First International ICSC Congress on Autonomous Intelligent Systems, ICAIS 2002, Geelong, Australia NAISO, ICSC-NAISO Academic Press, 02 2002. - online: <http: //www.deepc-auv.de/deepc/bibliothek/pdf/icais.pdf>

[LZ04] Liebezeit, Thomas ; Zerbe, Volker: *A Validation Environment for Mission Level Design of Autonomous Underwater Vehicles*. In: Proceedings of the 49. Internationales Wissenschaftliches Kolloguium, Ilmenau, Germany, 2004

[LZL03] Liebezeit, Thomas ; Zerbe, Volker ; Löffler, Tino: *A Framework for Mission Level Design*. In: Proceedings of the Twelfth IASTED International Conference on Applied Simulation and Modelling, ASM 2003, Marbella, Spain, 2003. - online: <http://sahara.theoinf.tu-ilmenau.de/research/ publications/documents/2003/asm2003.pdf>

[MLD03] MLDesigner Technologies, Inc. *MLDesigner Product Homepage*. online: <http://www.mldesigner.com>. 2003 8

[Sch00] Schorcht, Gunar: *Entwurf integrierter Mobilkommunikationssysteme auf Missionsebene*. Ed. 1. Berlin : Logos Verlag, 2000 3

[ZRL01] Zerbe, Volker ; Radtke, Torsten ; Liebezeit, Thomas: *Mission Level Design in Robotics*. In: Proceedings of 10th International Workshop on Robotics in Alpe-Adria-Danube region, RAAD'01, Vienna, Austria, 2001

7 Voraussichtlicher Nutzen, insbesondere Verwertbarkeit der Ergebnisse im Sinne des fortgeschriebenen Verwertungsplanes

Die Markterhebungen verschiedener Studien und europäischer Forschungsvorhaben zeigen, dass die Entwicklung selbstangetriebener, unbemannter, autonom operierender Unterwasserfahrzeuge weltweit immer mehr an Bedeutung gewinnt. Dies liegt darin begründet, dass sie gegenüber der aktuellen ROV-Technologie ein erheblich größeres Potential hinsichtlich Kostenminimierung, Effektivität und Handhabung aufweisen.
Der vorteilhafte Einsatz von AUVs wird auch aus der Sicht von ROV-Betreibern bestätigt.

Die in dem Teilthema prädiktives Führungssystem vorgeschlagenen neuen Methoden für ein intelligentes Verhalten von AUVs bzw. für die Unterstützung des gesamten Hard- und Softwareentwurfs konnten erfolgreich gelöst werden.

Bei ersten Flachwassertests in der Ostsee konnte die Funktionsfähigkeit der neu entwickelten Methoden beim Einsatz in einem anderen Testfahrzeug nachgewiesen werden.

- Die am Projekt DeepC beteiligten Mitarbeiter:
  Dipl.-Ing. Mike Eichhorn
  Dipl.-Ing. Thorsten Pfützenreuter
  Dipl.-Ing. Thomas Liebezeit
  Dipl.-Ing. Divas Karimanzira
  nutzten die erzielten Ergebnisse für ihre Graduierungsarbeiten zum Dr.-Ing.
  Herr Karimanzira hat seine Dissertation inzwischen erfolgreich verteidigt, während sich die drei anderen Mitarbeiter in der Endphase befinden und ihre Dissertationen in nächster Zeit fertig stellen werden.

- In das Projekt DeepC waren insgesamt 12 wissenschaftliche Hilfskräfte integriert, deren Arbeiten ebenfalls ihrer wissenschaftlichen Qualifizierung dienten.

- Ausgewählte Ergebnisse des Projektes werden sukzessive in Lehrveranstaltungen des Fachgebietes Systemanalyse einbezogen und verbessern somit die Qualität der studentischen Ausbildung.

- Geeignete Technologien des DeepC-Projektes werden in anderen Bereichen der Grundlagenforschung des Fachgebietes Systemanalyse zum Entwurf und der Führung mobiler Systeme angewendet.

- Die im Zusammenhang mit dem Projekt DeepC entstandenen zahlreichen Veröffentlichungen sind in Abschnitt 9. aufgelistet.

8 Während der Durchführung des Vorhabens bekannt gewordene Fortschritte auf dem Gebiet des Vorhabens bei anderen Stellen

Es sind keine Fortschritte bei anderen Stellen bekannt geworden.

9 Erfolgte oder geplante Veröffentlichungen

Eichhorn, M.: *Control Tasks in the Development of Underwater Vehicle*, NRC IMD Seminars, Institute for Marine Dynamics, National Research Council Canada, St. John’s, Newfoundland, Canada, November 12, 2002.
Eichhorn, Mike: *Methoden zur Führung von unbemannten Unterwasserfahrzeugen*, 37. Regelungstechnisches Kolloquium, Boppard 19. - 21. Februar 2003.
Eichhorn, M.: *An Obstacle Avoidance System for an Autonomous Underwater Vehicle*.- In: Proceedings of the International Symposium on Underwater Technology, Taipei, Taiwan, April 20-23, 2004.
Eichhorn, M.: *An Obstacle Avoidance System for an Autonomous Underwater Vehicle*, 49. Internationales Wissenschaftliches Kolloquium Technische Universität Ilmenau, 27.-30. September 2004.
Eichhorn, M.: *A Control System for an Autonomous Underwater Vehicle in Special Situations*, 4 th Biannual NRC-IOT Workshop on Underwater Vehicle Technology, National Research Council Canada, Institute for Ocean Technology, St. John’s, Newfoundland, Canada, 14. - 15. October 2004.

Eichhorn, M.: *Ein Hindernisvermeidungssystem für Autonome Unterwasserfahrzeuge*, at – Automatisierungstechnik, Volume 52, Issue 11, pp. 514-525 2004.
Pfützenreuter, T.: *An Onboard Mission Replanning System for Autonomous Underwater Vehicles*, 11th Mediterranean Conference on Control and Automation, Rhodos, Griechenland, 17. – 20. Juni 2003.
Pfützenreuter, T.: *Advanced Mission Management for Long-range Autonomous Underwater Vehicles*, MTS/IEEE Oceans 2003, San Diego, USA, 22. – 26. September 2003.
Pfützenreuter, T.: *Intelligent Mission Management for Autonomous Mobile Systems*, 49. Internationales Wissenschaftliches Kolloquium, Ilmenau, 27. - 30. September 2004.
Otto, P.: *Machine Learning Fuzzy Method for the Modelling of Experts and Systems*
Proc. of the 2003 European Simulation and Modelling Conference ESM 2003, pp. 237-241, October 27-29, Naples, Italy.
Karimanzira D. and Otto, P. (2004). *A Manoeuvre Management System Based On Machine Learning For An Autonomous Underwater Vehicle*, 49. Internationales Wissenschaftliches Kolloquium, Ilmenau, 27. - 30. September 2004.
Andrejevi'c, Miona ; Liebezeit, Thomas: *Modeling of the energy system for the mission level design of the DeepC AUV*. In: Proceedings of the twelfth international scientific and applied science conference ELECTRONICS, ET'2003, Sozopol, Bulgaria, 2003
Liebezeit, Thomas: *Mission Level Design of Autonomous Underwater Vehicles*. In: Nahavandi, Saeid (Hrsg.): Proceedings of the First International ICSC Congress on Autonomous Intelligent Systems, ICAIS 2002, Geelong, Australia NAISO, ICSC-NAISO Academic Press, 02 2002. - online: <http: //www.deepc-auv.de/deepc/bibliothek/pdf/icais.pdf>
Liebezeit, Thomas ; Zerbe, Volker: *A Validation Environment for Mission Level Design of Autonomous Underwater Vehicles*. In: Proceedings of the 49. Internationales Wissenschaftliches Kolloquium, Ilmenau, 27. - 30. September 2004.
Liebezeit, Thomas ; Zerbe, Volker ; Löffler, Tino: *A Framework for Mission Level Design*. In: Proceedings of the Twelfth IASTED International Conference on Applied Simulation and Modelling, ASM 2003, Marbella, Spain, 2003. - online: <http://sahara.theoinf.tu-ilmenau.de/research/ publications/documents/2003/asm2003.pdf>
Zerbe, Volker ; Radtke, Torsten ; Liebezeit, Thomas: *Mission Level Design in Robotics*. In: Proceedings of 10th International Workshop on Robotics in Alpe-Adria-Danube region, RAAD'01, Vienna, Austria, 2001

Geplante Veröffentlichungen:

Eichhorn, Mike: *A Reactive Obstacle Avoidance System for an Autonomous Underwater Vehicle*, eingereichter Vortrag zum 16th IFAC World Congress, Praga, Czech Republic, 4-8 July 2005
Pfützenreuter, T: *Intelligent Mission Replanning for Autonomous Mobile Systems*, eingereichter Vortrag zum 16th IFAC World Congress, Praga, Czech Republic, 4-8 July 2005
Otto, P.: *Machine Learning of Expert Decision or System Behaviour,* eingereichter Vortrag zum 16th IFAC World Congress, Praga, Czech Republic, 4-8 July 2005