UNIVERSITÄT OSNABRÜCK

Institut für Informatik
AG Wissensbasierte Systeme

# Prozessoptimierte Planung für kooperative mobile Roboter

*Dissertation*

zur Erlangung des Doktorgrades (Dr. rer. nat.)
des Fachbereichs Mathematik/Informatik
der Universität Osnabrück

vorgelegt von

*Stephan Scheuren*

März 2014

| | |
|---|---|
| Erstgutachter: | Prof. Dr. rer. nat. Joachim Hertzberg |
| Zweitgutachter: | Prof. Dr. rer. nat. Jürgen Richter |

## Zusammenfassung

In vielen industriellen Anwendungen kooperieren mobile Maschinen, um ein gemeinsames Prozessziel zu erreichen. Durch den Transfer von Technologien und Verfahren aus der Robotik agieren einzelne mobile Maschinen in vielen Bereichen bereits (teil)autonom. Der nächste Schritt der Automatisierung erfordert die Koordination von robotischen Systemen zur Erreichung bestimmter gemeinsamer Prozessziele. Um die Kooperation von roboterisierten mobilen Maschinen oder von mobilen Robotern allgemein zu ermöglichen, müssen ihre Pläne raumzeitlich aufeinander abgestimmt werden. In dieser Arbeit wird ein Planungssystem für ein kooperatives Ernteszenario aus dem Anwendungsbereich der Infield-Transportlogistik vorgestellt. In dem Szenario agieren ein Mähdrescher und ein Transportfahrzeug kooperativ, um den Ernte- und Transportprozess effizient ausführen zu können. Ein maschinenübergreifender Planungsschritt optimiert die Logistik des Gesamtprozesses. Ein weiterer, maschinenspezifischer Planungsschritt generiert aus dem gemeinsamen Prozessplan aller Fahrzeuge einen ausführbaren Teilplan für jedes einzelne Feldfahrzeug. Die Pläne werden aufgrund der Online-Problematik des Ernteszenarios während der Prozessausführung bei Bedarf dynamisch neu berechnet. Diese Arbeit zeigt die Umsetzbarkeit raumzeitlicher Planungssysteme für kooperative mobile Roboter in realen Anwendungsszenarien.



## Abstract

In many industrial applications mobile machines cooperate to accomplish a common process. By transfer of robotic technologies and methods, single machines already operate (semi-)autonomously in many areas. The next step towards automation requires coordination of robotic systems for achieving common process goals cooperatively. To enable the cooperation of mobile robotized machines or mobile robots in general, their plans have to be spatio-temporally coordinated. In this thesis, a planning system for a cooperative harvesting scenario emerging from the scope of infield-transport logistics is presented. In the scenario, one harvester and one transport vehicle operate cooperatively to efficiently accomplish the harvesting and transportation process of the crops. A high level planning step optimizes the common overall logistics plan for all vehicles. A subordinated, machine specific planning step generates executable subplans for the vehicles. Those plans are dynamically recalculated to deal with the online problematic of the harvesting scenario. This thesis demonstrates the feasibility of spatio-temporal planning systems for cooperative mobile robots in real-world applications.

# Danksagung

Bei Herrn Prof. Dr. Hertzberg bedanke ich mich für die Möglichkeit, in einem sehr angenehmen Arbeitsumfeld forschen und meine Dissertation über ein spannendes Thema verfassen zu können. Zudem danke ich ihm für die immer vorhandene Gesprächsbereitschaft und die gute Betreuung. Herrn Prof. Dr. Richter danke ich dafür, dass er mich als Zweitgutachter betreut hat. Darüber hinaus möchte ich mich bei Herrn Prof. Dr. Markus Chimani und Frau Prof. Dr. Sigrid Knust für die fachlichen Diskussionen bedanken.

Dem Bundesministeriums für Wirtschaft und Technologie (BMWi) gilt mein Dank für die Förderung des Projekts *marion*, in dem ich die letzten drei Jahre mitarbeiten durfte. Den Projektpartnern danke ich für die erfolgreiche Zusammenarbeit.

Allen meinen Arbeitskollegen am DFKI in Osnabrück und den Mitarbeitern der Arbeitsgruppe Wissensbasierte Systeme der Universität Osnabrück danke ich für die schöne gemeinsame Zeit und die vielen Diskussionen. Dr. Ronny Hartanto und Dr. Stefan Stiene danke ich für den kreativen Input bei offenen Problemen. Danke auch an Astrid Ullrich, Kai Lingemann und Sebastian Stock, die dafür gesorgt haben, dass es im Büro nie langweilig wurde.

Mein größter Dank gilt meiner Familie, insbesondere meinen Eltern, die mich stets unterstützt haben. Danke an David und Hannah dafür, dass sie mich nicht zu oft haben auf den Kopf fallen lassen und an Fabian dafür, dass er der zweitbeste Zwilling der Welt ist - oder habe ich uns jetzt verwechselt?

Last but not least möchte ich Jasmin dafür danken, dass sie seit so langer Zeit immer für mich da ist, mich inspiriert, motiviert und bedingungslos unterstützt. Ich liebe Dich.

# Inhaltsverzeichnis

# Abbildungsverzeichnis

# Kapitel 1

# Einleitung

In vielen industriellen und landwirtschaftlichen Prozessen kooperieren heute mobile Maschinen, um ein gemeinsames Prozessziel zu erreichen. Für diese Kooperation müssen oft raumzeitliche Treffpunkte abgestimmt und eingehalten werden. Ein industrielles Beispiel dafür sind Hochregallager, in denen Gabelstapler Schleppzüge be- und entladen. In der Landwirtschaft müssen beispielsweise beim Mähdrusch Erntefahrzeuge, Überladefahrzeuge und Straßentransporter die Logistik des Ernteguts gemeinsam bewerkstelligen.

Mit steigender Autonomie der beteiligten Maschinen müssen komplexere Planungssysteme realisiert werden, die das Zusammenspiel der Maschinen koordinieren. Um eine prozessoptimale Planung zu ermöglichen, müssen die Pläne der einzelnen Maschinen im Kontext des Gesamtprozesses aufeinander abgestimmt werden. Zur Autonomiesteigerung der Maschinen wird dabei häufig auf wissenschaftliche Erkenntnisse aus der Robotik zurückgegriffen.

Über die letzten Jahrzehnte haben daher Technologien und Verfahren aus der Robotik alltägliche und industrielle Anwendungen und Produkte verändert. Dieser Transfer ist nicht immer transparent, weil er oft auf funktionaler Ebene stattfindet. Ein prominentes Beispiel dafür sind Parklenkassistenten in modernen Automobilen, die Parklücken vermessen und beim Einparken die Lenkmanöver vollständig übernehmen. Die Technologie und die grundlegenden Algorithmen, die in diesem Produkt stecken, stammen aus der Robotik. Automobile sind längst zu komplexen mobilen Sensorplattformen geworden, die ihre Umwelt erfassen und in dieser (teil-)autonom agieren oder den Fahrer unterstützen können. An diesem Beispiel wird deutlich, dass Anwendern nicht immer bewusst wird, dass sie Technologien nutzen, die ihren Ursprung in der Robotik haben. In industriellen Anwendungen ist dies häufig transparenter, beispielsweise bei in der Produktion eingesetzten Schweißrobotern.

In dem Forschungsprojekt *marion* wurden robotische Technologien und Verfahren in zwei Anwendungsfälle transferiert, in denen mobile Maschinen kooperativ in einer Prozesskette agieren. In dieser Arbeit wird ein entwickeltes Planungssystem für den landwirtschaftlichen Anwendungsfall der Getreideernte mit Ernte- und Transportfahrzeugen beschrieben. Zielsetzung ist vor allem die Effizienzsteigerung des Gesamtprozesses durch eine prozessoptimierte Planung der koopera-

tiven Arbeitsschritte.

In den folgenden Abschnitten der Einleitung wird die Zielsetzung, der Aufbau und der wissenschaftliche Beitrag dieser Arbeit beschrieben.

## 1.1 Zielsetzung

Diese Arbeit beschreibt ein entwickeltes Planungssystem für kooperative Ernteprozesse, das im Projekt *marion* eingesetzt wurde. Wichtige und grundlegende Ziele dafür werden im Folgenden vorgestellt. Die Hauptziele der Arbeit sind mit * markiert.

**Formalisierung*:** Das landwirtschaftliche Szenario muss formal beschrieben werden, damit der Prozess der kooperativen Getreideernte geplant werden kann. Die gängige landwirtschaftliche Praxis soll Grundlage für die Planung sein.

**Raumzeitliche Planung*:** Damit die Feldfahrzeuge in dem Ernteprozess kooperieren können, müssen sie sich zu einem bestimmten Zeitpunkt an einem bestimmten Ort treffen. Für die Abstimmung dieser Treffpunkte ist entsprechend ein Verfahren zur Lösung des entstehenden raumzeitlichen Planungsproblems zu entwickeln.

**Prozessoptimierung*:** Wie einleitend erwähnt, ist unter anderem die Steigerung der Effizienz des Gesamtprozesses Motivation des Projekts. Daher ist ein Verfahren zu entwickeln, das eine Optimierung des Gesamtprozesses erlaubt.

**Dynamische Plananpassung:** Bei der Prozessausführung ist mit Abweichungen von dem Plan zu rechnen, weil in Ernteprozessen naturgemäß schwankende Erträge auftreten. Dadurch können Korntankfüllstände und Geschwindigkeiten der Erntefahrzeuge stark von Prognosen abweichen. Um diesen Abweichungen zu begegnen ist eine dynamische Plananpassung während der Prozessausführung umzusetzen. Diese Plananpassung soll auf einem Fahrzeugrechner und mit praktikabler Laufzeit ausgeführt werden können.

**Konsistenz der Teilpläne:** Die Pläne der Feldfahrzeuge müssen einem konsistenten gemeinsamen Prozessplan folgen, damit die genannten raumzeitlichen Treffpunkte kooperative Prozessabschnitte zulassen. Es ist also sicherzustellen, dass zu jedem Zeitpunkt immer nur ein maschinenübergreifender Plan des Ernteprozesses vorliegt und dass dieser auch von allen Feldfahrzeugen ausgeführt wird.

**Integration und Tests:** Das Planungssystem soll auf realen Maschinen integriert werden. Dazu muss eine Anbindung der zu entwickelnden Planungssoftware an die Maschinensteuerung realisiert werden. Das Planungssystem soll nach Möglichkeit in realen Ernteszenarien getestet werden.

## 1.2 Aufbau der Arbeit

**Kapitel 1** gibt eine Einleitung in das Thema der Arbeit. Der Aufbau und die wissenschaftlichen Beiträge der Arbeit werden erläutert.

**Kapitel 2** stellt das Projekt *marion* im Allgemeinen und das landwirtschaftliche Szenario des Mähdruschs mit dem daraus resultierenden Planungsproblem im Speziellen vor.

**Kapitel 3** beschreibt den Stand der Technik zur Planung für kooperative mobile Roboter und zeigt den Ansatz des entwickelten Planungssystems. Die weitere Unterteilung in Routen- und Bewegungsplanung wird beschrieben.

**Kapitel 4** erläutert die übergeordnete Routenplanung für die beteiligten Maschinen.

**Kapitel 5** beschreibt die Bewegungsplanung für die Maschinen in Abhängigkeit zu Maschinentyp und Prozesszustand.

**Kapitel 6** zeigt Experimente zur Routen- und Bewegungsplanung.

**Kapitel 7** fasst die Ergebnisse der Arbeit zusammen und gibt einen Ausblick auf weiterführende Arbeiten.

## 1.3 Wissenschaftlicher Beitrag

**Formalisierung des Planungsproblems:** Das Szenario der kooperativen Getreideernte wird in dieser Arbeit detailliert dargestellt. Das aus dem Szenario hervorgehende grundlegende Problem der raumzeitlich eingeschränkten Bewegungsplanung wird vorgestellt. Diese Beschreibung wird weiter vertieft und formal als Problem der maschinenübergreifenden Planung mit allen prozessbedingten Einschränkungen formuliert. Diese Formalisierung ermöglicht die Implementierung von Optimierungs- und Planungsverfahren und den Vergleich mit dem in dieser Arbeit vorgestellten Ansatz.

**Raumzeitliche Planung:** Es wird ein Verfahren zur raumzeitlichen Planung für kooperative mobile Roboter am Beispiel des Anwendungsfalls der kooperativen Getreideernte vorgestellt. Dieses Verfahren ist mit Anpassungen funktional auf andere Anwendungsfälle übertragbar.

**Bewegungsplanung:** Für Ernte- und Transportfahrzeuge werden fahrzeugtypspezifische Verfahren zur Bewegungsplanung für die kooperative Getreideernte beschrieben. Diese Verfahren basieren auf der frei verfügbaren Bibliothek *Search-Based Planning Library* [58] und auf einem im Rahmen dieser Arbeit entwickelten Verfahren zur Approximation einer Referenztrajektorie mit Bewegungsprimitiven innerhalb eines Toleranzbereiches.

**Prototypische Umsetzung auf realen Maschinen:** Mit der Beschreibung des entwickelten und prototypisch auf realen Maschinen integrierten Planungssystems sowie erfolgreich durchgeführten Erntetests wird die Umsetzbarkeit von raumzeitlichen Planungssystemen in komplexen Wertschöpfungsketten gezeigt.

# Kapitel 2

# Das Projekt *marion*

In diesem Kapitel wird das Projekt *marion* mit seinen beiden Anwendungsfällen Intralogistik und Infield-Transportlogistik beschrieben. Anschließend wird das Planungsproblem für den in dieser Arbeit betrachteten Anwendungsfall der Infield-Transportlogistik detailliert dargestellt.

*marion* steht für *mobile autonome, kooperative Roboter in komplexen Wertschöpfungsketten.* Das Konsortium setzte sich zusammen aus den Industriepartnern CLAAS und STILL und den Forschungspartnern Atos C-Lab und dem Robotics Innovation Center des Deutschen Forschungszentrum für Künstliche Intelligenz (DFKI RIC).

Ziel des Projekts war die Roboterisierung von Prozessen, in denen mobile Maschinen kooperativ agieren. Dies sollte die Autonomie der mobilen Maschinen erhöhen und die Effizienz der Prozesse steigern. Dafür wurden in den Anwendungsdomänen der beiden Industriepartner zwei Szenarien ausgewählt, für die ein Transfer von Technologien und Verfahren aus der Robotik geleistet werden sollte (siehe Abbildung 2.1).

In der Domäne der Intralogistik war die Aufgabenstellung der Transport von Stückgütern mit Hilfe von Gabelstaplern und Schleppzügen. Für die Planung der Kooperation der mobilen Maschinen war in diesem Anwendungsfall bereits ein Lagerverwaltungssystem (LVS) vorhanden. Trotzdem mussten die Fahrzeuge ihre Bewegungen selbstständig und dynamisch planen können. Deshalb wurde eine Planungsebene zur Bewegungsplanung durch das DFKI realisiert. Diese basiert mit der Search-Based Planning Library (SBPL) auf einer Bibliothek nach aktuellem Stand der Technik. Die Planausführung durch die Fahrzeuge hat in diesem Anwendungsfall der Industriepartner STILL verantwortet. Zusätzlich wurde in dem Anwendungsfall der Intralogistik eine Auswertung von dreidimensionalen Sensordaten der Umgebung benötigt. Diese Auswertung wird im Folgenden als Szenenanalyse bezeichnet. In den Sensordaten mussten Positionen von Anhängern und Ladungsträgern genau bestimmt werden, damit das Be- und Entladen von Paletten autonom durchgeführt werden konnte. Diese Szenenanalyse wurde ebenfalls vom Forschungspartner DFKI realisiert. Der Anwendungsfall der Intralogistik wird in Abschnitt 2.1 beschrieben.

Aus der landwirtschaftlichen Domäne wurde die Infield-Transportlogistik als Anwendungsfall

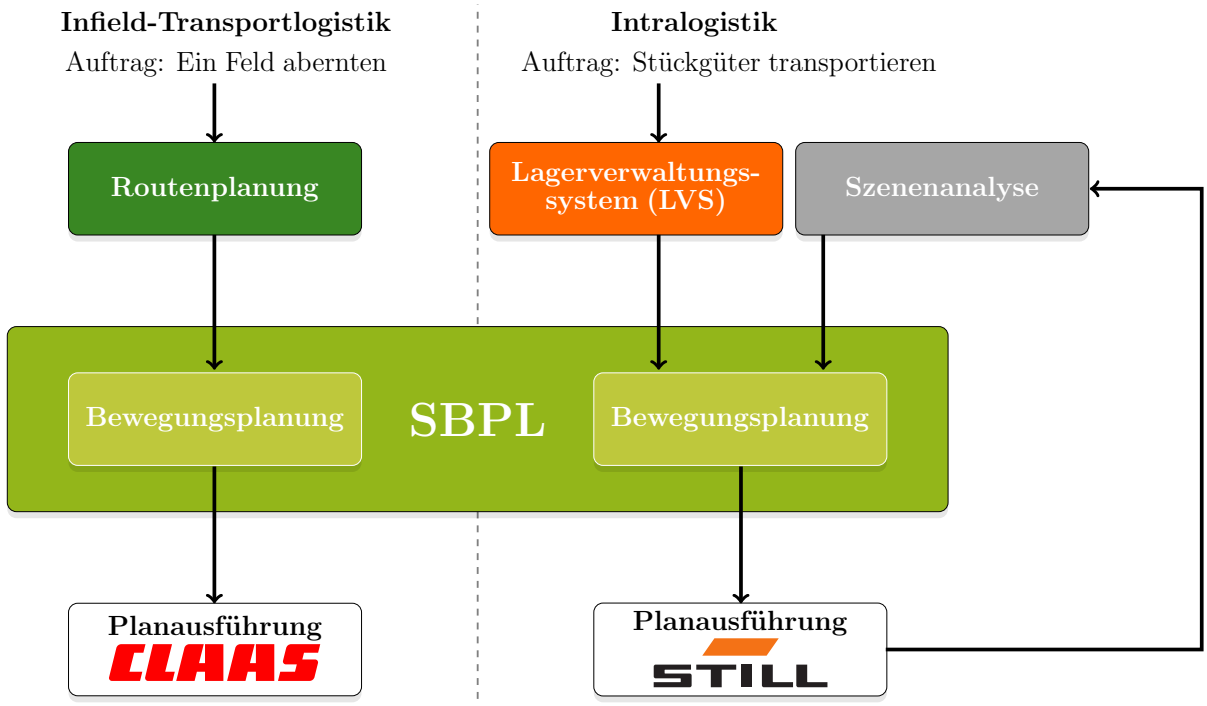



**Abbildung 2.1:** Übersicht über das Planungssystem in *marion* [41]. In beiden Szenarien musste eine Bewegungsplanung durchgeführt werden. Diese basierte in beiden Fällen auf der *Search-Based Planning Library (SBPL)* Bibliothek. Für die Infield-Transportlogistik wurde eine übergeordnete Planung für die Koordination der Maschinen benötigt. In der Intralogistik konnte auf ein vorhandenes Lagerverwaltungssystem zurückgegriffen werden. Neben der Bewegungsplanung war hier zusätzlich eine Szenenanalyse erforderlich, mit deren Hilfe genaue Standorte von Paletten und Schleppzügen bestimmt werden konnten.

ausgewählt. Die Aufgabe in diesem Szenario ist das Ernten und der Transport von Erntegut zum Feldrand. Zum Erfüllen dieser Aufgabe müssen Ernte- und Transportfahrzeuge gemeinsam agieren. Um eine Kooperation zu ermöglichen, sollte ein maschinenübergreifendes Planungssystem einen gemeinsamen Plan für alle beteiligten Fahrzeuge erstellen. Diese übergeordnete Planungsebene wird in dem Planungssystem als Routenplanung bezeichnet (Abbildung 2.1). Damit die beteiligten Fahrzeuge einem gemeinsamen Routenplan autonom folgen können, müssen sie ihre Bewegungen selbstständig und dynamisch planen können. Für diese Aufgabe wurde die Bewegungsplanung als weitere, der Routenplanung nachgeschaltete Planungsschicht implementiert (Abbildung 2.1). Wie auch für den Anwendugsfall der Intralogistik basiert diese auf der SBPL. Diese beiden Planungsschichten wurden durch das DFKI realisiert. Für die Planausführung durch die Maschinen war der Industriepartner CLAAS verantwortlich. Das Szenario der Infield-Transportlogistik wird in Abschnitt 2.2 genauer beschrieben.

## 2.1 Anwendungsfall Intralogistik

In diesem Anwendungsfall wurde der innerbetriebliche Warenverkehr von Stückgütern robotorisiert. Dabei stand die Kooperation von Schleppzügen für den Horizontaltransport und Gabelstaplern für den Vertikaltransport im Vordergrund. Dafür war unter anderem das autonome Be- und Entladen der Schleppzüge erforderlich. Als Fahrzeuge wurden der CTX Schleppzug und der FMX-autonom Gabelstapler des Partners STILL verwendet (Abbildung 2.2).


(a) Gabelstapler STILL FMX-autonom (b) Schleppzug STILL CXT

**Abbildung 2.2:** Fahrzeuge in dem Anwendugsfall Intralogistik [93]. Es wurden Schleppzüge des Typs CXT und Gabelstapler des Typs FMX-autonom des Partners STILL verwendet. In der Intralogistik werden Gabelstapler vor allem für den Vertikaltransport, Schleppzüge für den Horizontaltransport von Stückgütern eingesetzt. Durch Kooperation der Fahrzeugtypen kann ein effizienter Transport realisiert werden.

Abbildung 2.3 zeigt das Szenario aus der Intralogistik in dem Projekt *marion*. Die Schleppzüge bewegen sich auf vorgegebenen Bahnen, die verschiedene Arbeitsbereiche miteinander verbinden. Innerhalb dieser Arbeitsbereiche befinden sich die Regale oder Abstellplätze für die Paletten. Die autonomen Gabelstapler sollen in diesem Arbeitsbereich einen Ladungsträger an eine bestimmte Position einlagern oder einen Ladungsträger aus diesem Bereich aufnehmen und auf einem Anhänger eines Schleppzuges abladen. In dem Arbeitsbereich muss ein Gabelstapler dabei Hindernisse erkennen und diese durch dynamische Fahrwegsplanung umfahren können. Außerhalb der Arbeitsbereiche nutzen die Gabelstapler ebenfalls das statische Wegenetz.

Eine wichtige Aufgabe in diesem Szenario war eine Analyse der Umgebung durch die Fahrzeuge, die sogenannte Szenenanalyse (Abbildung 2.3). Ziel der Analyse war es, in dreidimensionalen Punktwolken, die kontinuierlich aus Laserscans der Umgebung erstellt wurden, die genaue Po-

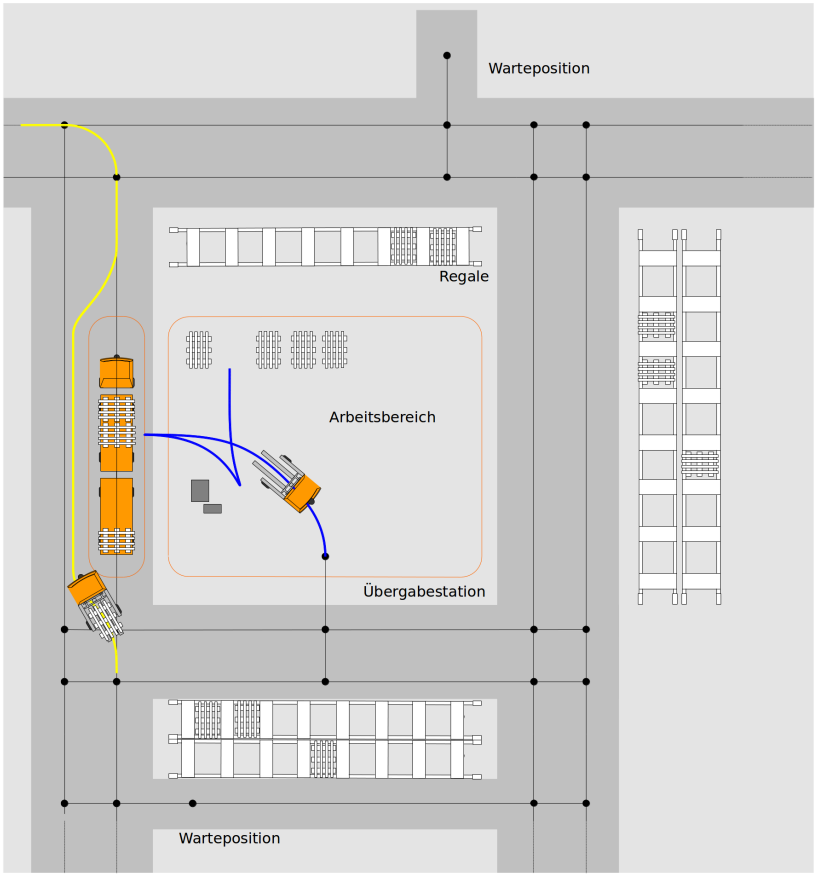



**Abbildung 2.3:** Szenario im Anwendungsfall Intralogistik [41]. Außerhalb der Arbeitsbereiche wurde ein statisches Wegenetz (schwarze Linien) für die Fortbewegung genutzt. Innerhalb der Arbeitsbereiche waren die Bewegungen (blaue Linie) des Gabelstaplers zu planen. Die Bewegungsplanung zur Umfahrung von Hindernissen außerhalb des Arbeitsbereichs (gelbe Linie) war ein weiterer Arbeitspunkt.

sition einer Palette oder eines Anhängers zu bestimmen. So konnte eine Fehlertoleranz in der Positionierung von Paletten und Fahrzeugen sichergestellt werden, die besonders im gemischten Betrieb von autonomen und von Menschen gesteuerten Fahrzeugen nötig ist.

In diesem Anwendungsfall wurde auf bestehende Architektur und Infrastruktur des Partners STILL zurückgegriffen. Der Gabelstapler FMX-autonom war zu Projektbeginn bereits mit Sensorik zur Umfelderkennung ausgestattet und für den gemischt autonomen Betrieb mit einer Personenschutzanalge ausgestattet. Die Verfahren zur Szenenanalyse und Bewegungsplanung konnten direkt in die modulare Architektur der Steuerungssoftware eingebunden werden. Die drahtlose Kommunikation zwischen den Fahrzeugen stand ebenfalls bereits vor Projektbeginn zur Verfügung.

## 2.2 Anwendungsfall Infield-Transportlogistik

In diesem Anwendungsfall wurde die Getreideernte und der Abtransport von Erntegut als logistischer Prozess geplant und automatisiert. Die Betrachtung der Logistikkette war beschränkt auf die Infield-Transportlogistik, also auf die Transportlogistik bis zum Feldrand. Als Feldfahrzeuge wurden dazu in dem Projekt *marion* ein Mähdrescher des Typs CLAAS LEXION 760 als Erntemaschine und ein Schlepper des Typs CLAAS XERION mit einem Überladewagen der Firma HAWE als Überladefahrzeug eingesetzt (siehe Abbildung 2.4). Am Feldrand befanden sich zusätzlich Straßentransporter, die den weiteren hier nicht betrachteten Transport des Ernteguts übernommen haben.

**Abbildung 2.4:** Fahrzeuge in dem Anwendugsfall Infield-Transportlogistik [25]. Es wurde ein Mähdrescher des Typs CLAAS LEXION 760 als Erntefahrzeug (links) und ein Schlepper des Typs CLAAS XERION mit einem Überladewagen von HAWE als Überladefahrzeug (rechts) eingesetzt. Das Bild zeigt einen Überladevorgang in Parallelfahrt.

In der Getreideernte ist es üblich, zu Beginn entweder die Stirnseiten oder die komplette jeweils äußere Kontur des Feldes mehrere Male erntend abzufahren. Dieser Prozessabschnitt wird als Freischneiden bezeichnet. Ergebnis des Freischneidens ist eine als Vorgewende bezeichnete Fläche, die für Wendemanöver der Feldfahrzeuge und für das Anfahren von Straßentransportern genutzt werden kann. In der Praxis gibt es verschiedene Strategien für das Freischneiden. In dem Projekt *marion* wurde eine Strategie fest vorgegeben. Bei dieser wurde die jeweils äußere Feldkontur abhängig von der Arbeitsbreite der Erntemaschine mehrere Male freigeschnitten, so dass die Breite des entstandenen Vorgewendes für das autonome Wenden der Maschinen ausreichte. Die Arbeitsbreite beschreibt die Breite, in der Getreide von einer Erntemaschine geerntet wird und berechnet sich als Differenz der tatsächlichen Schneidwerksbreite und einer vorgegebenen Überlappung.

Nach dem Freischneiden des Vorgewendes wurde die verbleibende innere Feldfläche in parallel verlaufende Fahrspuren unterteilt. Diese Fahrspuren deckten für die entsprechende Arbeitsbreite die gesamte abzuerntende Fläche ab. In der Praxis geschah das bislang nach Augenmaß der Fahrer. Hier wurden die Fahrspuren vorab berechnet und anschließend möglichst präzise durch die Erntefahrzeuge abgearbeitet. Währenddessen musste das Überladefahrzeug gewährleisten, dass die Korntanks der Erntefahrzeuge nie vollständig gefüllt waren, um den Stillstand der Erntefahrzeuge und damit den Verlust von wertvoller Prozesszeit zu vermeiden.

Das Entleeren der Korntanks der Erntefahrzeuge in den Überladewagen sollte dabei in Parallelfahrt zu dem Überladefahrzeug und während des Erntens geschehen, wie in Abbildung 2.4 dargestellt. Für das Überladen in Parallelfahrt schwenkt das Erntefahrzeug sein Korntankrohr aus und ein Überladefahrzeug positioniert sich relativ zu dem Erntefahrzeug so, dass sich der Auslauf des Korntankrohrs über seinem Anhänger befindet. Dabei war zu beachten, dass das Korntankrohr des eingesetzten Mähdreschers nur nach links ausgeschwenkt werden kann. Das Überladefahrzeug musste sich daher bei der Parallelfahrt und dem Überladen immer in Fahrtrichtung links von dem Mähdrescher befinden. Wenn die entsprechende Positionierung erreicht wurde, so konnte der Überladeprozess durch das Einschalten der Korntank-Entleerschnecke des Mähdreschers begonnen werden. Während des Überladens musste der Fahrer des Überladefahrzeugs Geschwindigkeit und Fahrweg des Erntefahrzeugs beachten, um die Position des Korntankrohr Auslaufs über dem Anhänger zu halten. Dabei musste er die Positionierung in Fahrtrichtung so variieren, dass der Anhänger gleichmäßig gefüllt wurde. Wenn ein Überladewagen voll war, musste der Fahrer des Überladefahrzeugs den am Feldrand wartenden Straßentransporter anfahren. Dort musste der Überladewagen entleert werden, bevor der nächste Mähdrescher zum erneuten Überladen angefahren werden konnte.

Zu Beginn des Projektes gab es für den Fahrer eines Überladefahrzeugs während des Ernteprozesses wenig Informationen als Entscheidungsgrundlage dafür, welcher Mähdrescher wann entladen werden sollte. In der Praxis wurde zurückgegriffen auf Funkkommunikation zwischen den Fahrern der Erntefahrzeuge und auf eine gelbe Rundumkennleuchte an den Erntefahrzeugen, die mit Hilfe eines Füllstandssensors ab einem Korntankfüllstand von 70% automatisch angeschaltet wurde.

Abbildung 2.5 zeigt schematisch das Szenario für den oben beschriebenen Prozess der Getreideernte mit Ernte- und Transportfahrzeugen im Projekt *marion*. Das Planungssystem sollte die Ausführung des Szenarios planen, überwachen und auf Abweichungen durch Plananpassungen reagieren. Der Freischneidevorgang sollte aufgrund von Hindernissen am Feldrand, teilweise nicht vorliegenden Feldkonturen und erhöhtem Manövrieraufwand weiterhin manuell ausgeführt werden. Im Anschluss an das Freischneiden des Vorgewendes sollte das Planungssystem den weiteren Prozessablauf planen. Dafür musste die innere Feldgrenze bestimmt werden, die die noch nicht abgeerntete Feldfläche umschloss (siehe Abbildung 2.5). Anschließend musste ein Satz von Fahrspuren berechnet werden, der diese Fläche für die gegebene Arbeitsbreite vollständig abdeckte. Für die Erntefahrzeuge musste dann eine Reihenfolge gewählt werden, bei der jede Fahrspur genau einmal abgeerntet wurde. Ausgehend von der Maschinen-, Prozesskonfigurationen und der Fahrspurreihenfolge der Erntefahrzeuge musste dann eine Abtankreihenfolge für das Überladefahrzeug gefunden werden, die gewährleistete, dass kein Erntefahrzeug den Ernteprozess

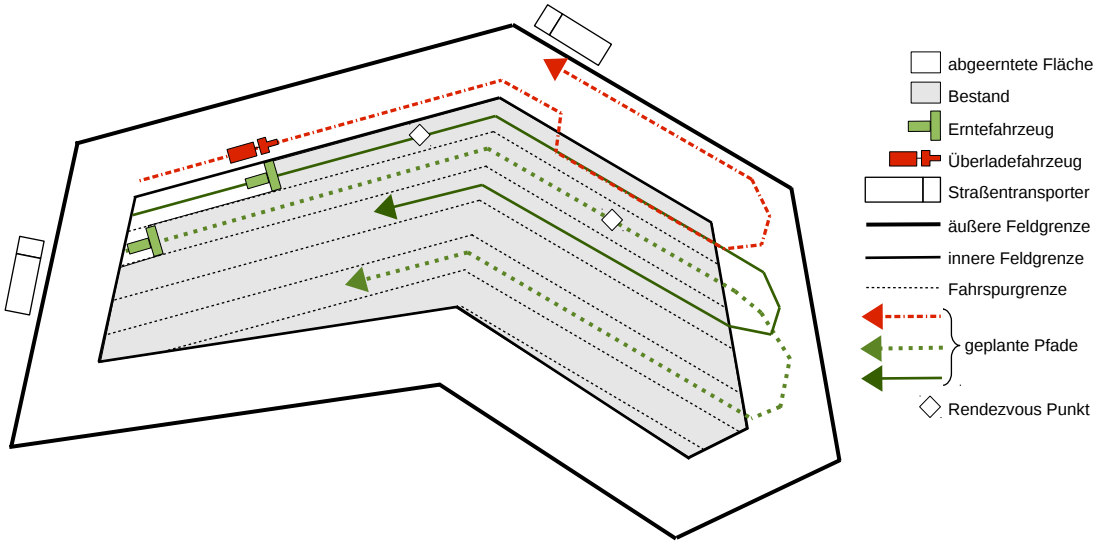



**Abbildung 2.5:** Szenario im Anwendungsfall Infield-Transportlogistik [87].

aufgrund eines vollständig gefüllten Korntanks unterbrechen musste. Dafür mussten Treffpunkte für den Start von Überladeprozeduren bestimmt werden. Diese Treffpunkte beinhalteten Vorgaben für Position, Orientierung, Zeit und Geschwindigkeit. In Abbildung 2.5 sind exemplarisch die ersten beiden Treffpunkte für die zwei Erntefahrzeuge und das Überladefahrzeug gegeben. In dem Beispiel wird erst das Erntefahrzeug auf der obersten Fahrspur entladen, während das Überladefahrzeug parallel dazu im Vorgewende fährt. Danach lässt sich das Überladefahrzeug zurückfallen, wechselt auf die oberste Fahrspur und fährt so auf bereits abgeernteter Fläche hinter dem gerade abgetankten Erntefahrzeug her. Dort wird der zweite Treffpunkt erreicht und das zweite Erntefahrzeug entladen. Anschließend fährt das Überladefahrzeug in dem Beispiel zum Feldrand, um den Inhalt des Überladewagens in einen Straßentransporter zu entleeren.

## 2.3 Planungsaufgabe der Infield-Transportlogistik

Aus der Szenariobeschreibung des Anwendungsfalls Infield-Transportlogistik aus Abschnitt 2.2 ergibt sich die Aufgabenstellung für das zu entwickelnde Planungssystem. Das zugrunde liegende Planungsproblem wird in [87] als *spatio-temporally constrained motion planning problem* (STCMP), also als raumzeitlich eingeschränktes Bewegungsplanungsproblem, mit folgenden Eigenschaften beschrieben.

Ausgehend von den geometrischen Parametern

- äußere Feldgrenze als Polygon
- innere Feldgrenze und statische Hindernisse als Polygon mit Löchern
- parallelen Fahrspuren, die die innere Feldfläche abdecken, als Polylinien

- Abtankorte für das Überladefahrzeug am Feldrand als Punkte

und fahrzeugspezifischen Parametern

- kinematische und dynamische Einschränkungen (Wenderadius, Beschleunigung, Geschwindigkeit)
- Korntankkapazität
- Entladeleistung
- Arbeitsbreite (nur für Erntefahrzeuge)
- maximale Ernteleistung (nur für Erntefahrzeuge)

und den Einschränkungen

- Erntefahrzeuge sollen während des Ernteprozesses nicht anhalten
- Erntefahrzeuge sollen innerhalb der inneren Feldfläche den Fahrspuren möglichst präzise folgen
- Erntefahrzeuge können nur zur linken Seite überladen
- Überladefahrzeug darf sich nur auf abgeernteter Fläche bewegen
- Überladefahrzeug soll möglichst die gleichen Fahrspuren nutzen, darf diese aber auch in niedriger Geschwindigkeit kreuzen
- die komplette innere Feldfläche soll abgeerntet werden

finde

- **Treffpunkte für die Kooperation** als Punkte, die durch Position, Orientierung, Zeitpunkt, Geschwindigkeiten und Überlademenge beschrieben werden
- **abfahrbare Pfade für alle Feldfahrzeuge** als Satz von Wegpunkten, die beschrieben werden durch Position, Orientierung, Zeitpunkt und Geschwindigkeit.

Die gegenseitige raumzeitliche Einschränkung der Fahrwege stellt eine Herausforderung für die Planung dar. Die Fahrzeuge haben außerdem gegenläufige Optimierungsziele: Das Überladefahrzeug hat als Optimierungsziel die Minimalisierung der maximalen Füllstände der Erntefahrzeuge über den gesamten Prozess, so dass möglichst kein Erntefahrzeug seinen Ernteprozess unterbrechen muss. Die Erntefahrzeuge hingegen sollen möglichst effizient und schnell arbeiten, was kurze Wege nahelegt. Das kann unter gewissen Umständen dazu führen, dass ein Erntefahrzeug stehen bleiben muss, etwa weil für das Überladefahrzeug kein rechtzeitiger Treffpunkt bestimmt werden kann, oder weil durch die Wahl der Fahrspur die zur Fahrtrichtung linke Fahrspur zum nötigen Zeitpunkt noch nicht abgeerntet wurde.

Zu dem Themenbereich der Aufgabenstellung gibt es in der Literatur einige Arbeiten, beispielsweise [3, 13–15, 35, 45]. Wie Abschnitt 3.1.6 erläutert, löst jedoch keine dieser Arbeiten die beschriebene Problemstellung.

# Kapitel 3

# Planung für kooperierende mobile Roboter

In diesem Kapitel werden die Grundlagen und der Stand der Technik der Planung für kooperierende mobile Roboter erläutert (Abschnitt 3.1). Im Anschluss werden in Abschnitt 3.2 die Voraussetzungen und Randbedingungen des Projekts *marion* beschrieben und die Architektur sowie der Framework des entwickelten Planungssystems für den Anwendungsfall der Infield-Transportlogistik vorgestellt.

## 3.1 Grundlagen und Stand der Technik

Planung in der Robotik beschäftigt sich mit dem generellen Problem, herauszufinden, welche Aktionen und Bewegungen ein Roboter in einer bestimmten Reihenfolge ausführen muss, um eine bestimmte Aufgabe zu erfüllen [100]. Es wird zwischen Handlungs- und Bewegungsplanung unterschieden. Die Handlungsplanung ist ein Teilgebiet der Künstlichen Intelligenz [83], das Methoden zum Schlussfolgern von Aktionen für das Erfüllen bestimmter Aufgaben erforscht. Grundlage für das Schlussfolgern ist eine spezifische Planungsdomäne mit einer Beschreibung des aktuellen und des zu erreichenden Zustands und den möglichen Aktionen und Bedingungen. In der Bewegungsplanung wird allgemein eine Trajektorie in einer Umgebungsrepräsentation gesucht, die einen Roboter von einer Start- in eine Zielkonfiguration überführt. Weil in dem Kontext des Infield-Transportlogistik Szenarios keine Handlungsplanung erforderlich ist, wird im Folgenden nur auf die Bewegungsplanung eingegangen.

Damit sich ein mobiler Roboter in seiner Umgebung bewegen kann, muss er seine Umgebung kennen, sich in dieser lokalisieren und eine Trajektorie zu einer Zielkonfiguration planen und ausführen können. Diese Schritte werden unter dem Begriff Navigation zusammengefasst [42]. Gängige Arten von Umgebungsrepräsentationen werden in Abschnitt 3.1.1 beschrieben. Abschnitt 3.1.2 zeigt Methoden zur Lokalisierung. In Abschnitt 3.1.3 wird ein Überblick von relevanten Verfahren zur Bewegungsplanung gegeben.

Bei der Planung des Infield-Transportlogistik Szenarios müssen die Bewegungspläne der Maschinen aufeinander abgestimmt werden, damit sie kooperative Arbeitsschritte gemeinsam durchführen können. Ein Überblick der Planung für kooperative Roboter wird in Abschnitt 3.1.4 gegeben. Abschnitt 3.1.5 erläutert einige Optimierungsverfahren, die für eine prozessoptimierte Planung des Gesamtprozesses in Frage kommen. Abschließend wird in Abschnitt 3.1.6 der Stand der Technik der Robotik in der Landwirtschaft vorgestellt.

### 3.1.1 Arten der Umgebungsrepräsentation

Die gängigen Arten der Umgebungsrepräsentation sind kontinuierlich metrische, diskret metrische und topologische Karten [42]. Kontinuierlich metrische Karten bilden die Umgebung exakt ab. Dadurch ermöglichen sie präzises Planen, erschweren aber durch ihre Kontinuität die Planung. Diskrete Karten reduzieren die Darstellung der Umgebung, zum Beispiel durch eine Rasterkarte. Diese Diskretisierung hat den Nachteil, dass die Umgebung nur annähernd repräsentiert werden kann, dafür aber den Vorteil, dass in dieser Darstellung effizientere Algorithmen angewendet werden können. Topologische Karten modellieren die Umgebung in einer Darstellung als Graph, so dass mit graphbasierten Suchverfahren kürzeste Pfade gesucht werden können. Topologische Karten haben im Allgemeinen den Nachteil, dass sie die Umgebung nur sehr grob beschreiben.

### 3.1.2 Lokalisierung

Es gibt relative und absolute Verfahren zur Lokalisierung [42, 100]. Relative Verfahren stützen sich auf Messwerte der Bewegungen des Roboters. Diese können durch Odometrie, beispielsweise durch die Messung von Radumdrehungen, oder inertiale Navigationssysteme bestimmt werden. Relative Verfahren kommen ohne a priori bekannte Umgebung aus. Absolute Verfahren zur Lokalisierung benötigen hingegen a priori Informationen zu Landmarken, aktiven Sendern oder zu kompletten Umgebungsmodellen, an denen sie sich orientieren [42]. Einen Sonderfall bildet das Verfahren des simultanen Lokalisierens und Kartierens (SLAM) [98], bei dem ein mobiler Roboter gleichzeitig eine Karte erstellt und in dieser seine Pose bestimmt.

Für mobile Roboter, die sich im Freien lokalisieren müssen, eignet sich das globale Positionsbestimmungssystem (GPS) [67]. Dabei handelt sich um ein absolutes Verfahren zur Lokalisierung, das Laufzeitmessungen von Signalen für die aktuelle Positionsberechnung nutzt. Um die Messgenauigkeit zu erhöhen, kann ein Korrektursignal von einer Referenzstation mit hochgenau eingemessener Position genutzt werden [67].

### 3.1.3 Bewegungsplanung

Die Bewegungsplanung ermöglicht mit der globalen Betrachtung einer Umgebung die vorausschauenste Art der Fortbewegung und der Hindernisvermeidung von mobilen Robotern. Neben ihr existierten simplere lokale Ansätze der reflexiven und reaktiven Fortbewegung. Ein Beispiel

für reflexive Fortbewegung sind die in [16] von Valentino Braitenberg beschriebenen Gedankenexperimente zu den nach ihm benannten Braitenberg-Vehikeln. Bei diesen Vehikeln werden Sensoren unter anderem direkt mit Aktoren verbunden. Trotz dieser reflexiven Kopplung kann an Braitenberg-Vehikeln komplexes Verhalten beobachtet werden [42]. Zu den reaktiven Methoden zur Fortbewegung gehören die Bug-Algorithmen, beispielsweise Bug1 und Bug2 [61]. Voraussetzung dieser Algorithmen ist, dass die Zielrichtung jederzeit bekannt ist. Ein Roboter bewegt sich in dieser Richtung, bis er auf ein Hindernis stößt. Ein Hindernis wird reaktiv umfahren und die Zielrichtung wird weiter verfolgt, bis das Ziel erreicht wird. Diese lokalen Ansätze sind für eine prozessoptimierte Planung ungeeignet, da nicht der Gesamtprozess betrachtet werden kann.

Die generelle Aufgabenstellung der Bewegungsplanung ist NP-vollständig [19]. Man unterscheidet bei der Bewegungsplanung zwischen dem Arbeitsraum und dem Konfigurationsraum [54]. Der Arbeitsraum beschreibt den physischen Raum, in dem sich ein Roboter befindet [42]. Der Konfigurationsraum hingegen beschreibt den Parameterraum, dessen Dimensionalität durch die Anzahl der Freiheitsgrade des Roboters bestimmt wird [24]. Er beschreibt alle Konfigurationen, die ein Roboter in einer Umgebung einnehmen kann. Der Konfigurationsraum wird aus der Transformation der Geometrien des Arbeitsraums und des Roboters unter Berücksichtigung kinematischer Einschränkungen des Roboters berechnet [54]. In dem berechneten Konfigurationsraum kann eine gültige, kollisionsfreie Folge von Konfigurationen von einer Start- zu einer Zielkonfiguration bestimmt werden. Diese kann in den Arbeitsraum rücktransformiert und durch den Roboter ausgeführt werden. Die Planung im Konfigurationsraum eignet sich vor allem für statische Umgebungen, da jede Änderung im Arbeitsraum eine Neuberechnung des Konfigurationsraums erforderlich macht. Deshalb wird in der Bewegungsplanung für mobile Roboter in dynamischen Umgebungen meist direkt im Arbeitsraum geplant. Dies ermöglicht kurze Laufzeiten, die für eine Reaktion auf dynamische Hindernisse erforderlich sind. Dabei ist es ein gängiges Verfahren, alle Hindernisse des Arbeitsraums um den halben Roboterradius zu vergrößern, um den Grundriss des Roboters als punktförmig annehmen zu können. So können mögliche Trajektorien bei der Planung effizient auf Kollisionen mit Hindernissen überprüft werden.

Klassische Verfahren Bewegungsplanung nutzen Straßenkarten, Raster- und Zellkarten, oder Potenzialfelder [42, 49, 54]. Darüber hinaus wurden Ansätze für spezielle Anforderungen und Randbedingungen entwickelt, etwa für die vollständig abdeckende Pfadplanung. Für diese Arbeit wird im Folgenden eine relevante Auswahl dieser Verfahren vorgestellt.

#### Bewegungsplanung mit Raster- und Zellkarten

Grundlage bildet bei diesen Verfahren zur Bewegungsplanung eine Raster- oder Zellkarte. Die Zerlegung in Zellen oder ein Raster geschieht exakt oder approximativ. Die exakte Zerlegung beschreibt die Umgebung verlustfrei, während die approximative Zerlegung die Umgebung diskretisiert und angenähert repräsentiert [54].

Bei der exakten Zellzerlegung wird der Freiraum einer polygonalen Umgebung durch trapezoide oder dreieckige Zellen beschrieben. Das bekannteste Verfahren zur exakten Zellzerlegung wurde in [22] vorgestellt und funktioniert nach dem Sweep-Gerade Prinzip [11]. Aus den Zellen wird ein Graph konstruiert, in dem zusammenhängende Zellen durch Kanten verbunden sind. Als

Knoten des Graphen können die Mittelpunkte der Zellen oder der gemeinsamen Kanten gewählt werden [65]. In dem resultierenden Graphen kann nach dem Einfügen von Knoten für Start- und Zielpose mit gängigen Suchalgorithmen der kürzeste Pfad bestimmt werden [54]. Aufgrund der verlustfreien Darstellung der Umgebung ist die Bewegungsplanung mit exakter Zellzerlegung vollständig [100].

Approximative Verfahren können den Freiraum in Rechtecke [6], uniforme Raster, oder Quadtrees [68] aufteilen. Die Zellen können frei, belegt, oder teilweise belegt sein. In der diskreten Darstellung einer approximierten Zellzerlegung kann der Pfad mit graphbasierten Suchalgorithmen bestimmt werden [49], zum Beispiel mit dem Dijkstra [26] oder dem A* Algorithmus [32].

#### Bewegungsplanung mit Potenzialfeldern

Bei diesen Verfahren werden für mögliche Zustände des Roboters, etwa für jede Zelle einer Rasterkarte, Potenziale berechnet. Grundlegende Idee ist, dass ein Roboter ausgehend von einer beliebigen Startposition dem Gradientenabstieg des Potenzials benachbarter Zellen bis zum Ziel folgen kann. Die Potenziale können auf verschiedene Arten berechnet werden. Mit der Aktivierungsausbreitung [32, 63] und dem Wavefront Algorithmus [68] können beispielsweise Potenziale anhand einer einfachen Breitensuche vom Ziel aus zu allen möglichen Startpositionen berechnet werden. Wenn nur die Distanz zum Ziel berücksichtigt wird, stellt bei dem Wavefront Algorithmus die Distanz einer Zelle direkt ihr Potenzial dar.

Potenzialfeldverfahren [42] sind verallgemeinert Ansätze, bei denen aus der Zielposition und den Hindernissen der Umgebung ein Potenzialfeld berechnet wird. Die Zielposition übt eine anziehende, die Hindernisse üben eine abstoßende Kraft aus. Das resultierende Potenzialfeld soll einen Roboter von einer beliebigen Startposition durch die wirkenden Kräfte zum Ziel führen. Ein Problem dieses Verfahrens sind lokale Minima in dem Potenzialfeld, die beispielsweise bei konkaven Hindernissen auftreten können [49]. Für das Verlassen dieser lokalen Minima gibt es einige Methoden, zum Beispiel durch Randomisierung [9].

#### Bewegungsplanung mit Straßenkarten

Bei der Bewegungsplanung mit Straßenkarten wird je nach Verfahren der Freiraum des Arbeits- oder des Konfigurationsraums als Graph in einer Straßenkarte abgebildet. Dieser Straßenkarte werden zusätzlich Start- und Zielkonfiguration für den Roboter hinzugefügt, so dass mit einer Graphsuche der kürzeste Weg bestimmt werden kann.

Für die Erzeugung der Straßenkarten existieren deterministische und probabilistische Ansätze. Ein bekanntes deterministisches Verfahren für die Konstruktion einer Straßenkarte ist der Sichtbarkeitsgraph [7]. Dabei werden alle untereinander sichtbaren Eckpunkte aller polygonalen Hindernisse miteinander verbunden. Untereinander sichtbar bedeutet, dass die direkte Verbindung zwischen den Punkten kein Hindernis schneidet. Eine Straßenkarte, die auf diese Art erstellt wird, ermöglicht eine vollständige Suche nach dem kürzesten Pfad [54]. Das bedeutet, dass immer die beste Lösung gefunden wird, falls es eine Lösung gibt. Ein anderes deterministisches,

ebenfalls geometrisches Verfahren nutzt einen Voronoi Graphen [20] als Straßenkarte. In diesem Graphen hat jede Kante denselben Abstand zu zwei, jeder Knoten denselben Abstand zu mindestens drei Eckpunkten von Hindernissen. Mit dem ersten Ansatz [7] wird der kürzeste Weg, mit dem zweiten Ansatz [20] ein sicherer Weg mit maximalem Abstand zu den Hindernissen ermittelt.

Bei probabilistischen Verfahren wird die Straßenkarte durch Abtasten des Freiraums des Arbeits- oder Konfigurationsraums erzeugt. Diese Verfahren können probabilistische Vollständigkeit [9] liefern. Das bedeutet, die Wahrscheinlichkeit dafür, dass ein Planer keine Lösung findet, obwohl eine existiert, tendiert für eine unendliche Anzahl an Abtastungen gegen Null [47]. Wenn die Abtastung im Arbeitsraum des Roboters durchgeführt wird, müssen abgetastete Zustände auf Hinderniskollisionen hin überprüft werden. Die probabilistischen Verfahren haben vor allem in komplexen und dynamischen Umgebungen gegenüber den deterministischen Verfahren den praxisrelevanten Vorteil, eine kürzere Laufzeit zu ermöglichen. Zu den bekanntesten probabilistischen Verfahren gehören Probabilistic RoadMaps (PRMs) [48, 50], Rapidly-exploring Random Trees (RRTs) [55, 56] und Expansive Search Trees (ESTs) [44]. Während die PRMs dabei Abfragen für verschiedene Start- und Endzustände erlauben, müssen bei RRTs und ESTs für jede Anfrage die Straßenkarten neu erstellt werden.

#### Vollständig abdeckende Pfadplanung

Diese vollständig abdeckende Pfadplanung sucht nach einem Weg für einen Roboter, der eine freie Fläche in einer Umgebung vollständig abdeckt. Anwendungen für diese Planung sind beispielsweise Säuberung, Minenräumung und Ernte [1, 23, 71]. Viele der Verfahren greifen für die Planung auf eine Zellzerlegung des Freiraums zurück [71]. Die Ansätze zur Zellzerlegung wurden bereits in dem Abschnitt zur Bewegungsplanung mit Raster- und Zellkarten erläutert. Der einfachste Ansatz ist die Bewegung nach dem Zufallsprinzip [23]. Choset argumentiert in seinem Beitrag [23], dass dieser Ansatz aufgrund von geringen Produktionskosten durch Einsparung von Sensorik und Rechenleistung kosteneffizient sein kann, obwohl er nicht vollständig und nicht optimal ist. Wie Oksanen in [71] feststellt, ist eine zufällige Abarbeitung eines Feldes offensichtlich nicht kosteneffizient, da die Maschinen- und Prozesskosten deutlich die Kosten für Sensorik und Rechenleistung übersteigen. Abgesehen von den fixen Kosten eines Roboters oder roboterisierten Feldfahrzeugs ist zudem die erhöhte Bodenbelastung durch mehrfaches Befahren derselben Fläche ein nicht zu vernachlässigendes Argument gegen zufällige Bewegungen.

Zur vollständig abdeckenden Pfadplanung kann ein Wavefront Algorithmus auf einer Rasterdarstellung des Freiraums angewendet werden [106]. Zwei weitere Ansätze werden von Oksanen und Visala in [71] vorgestellt. Der erste zerlegt eine Fläche in trapezoide Zellen und bestimmt für diese Zellen die besten Fahrtrichtungen. Der zweite Ansatz ist inkrementell und plant nur einen begrenzten Zeithorizont in Abhängigkeit des aktuellen Zustands und der abgearbeiteten Fläche.

Für das beschriebene Infield-Transportlogistik Szenario dieser Arbeit ist keiner der beschriebenen Ansätze geeignet, da die Feldfläche in parallel verlaufenden Fahrspuren abgearbeitet werden soll,

um das Überladen in Parallelfahrt planen und durchführen zu können. Daher werden geometrisch Fahrspuren erzeugt, die für eine gegebene Arbeitsbreite das Feld abdecken.

#### Planung für nicht-holonome mobile Roboter

Nicht-holonome Systeme können nicht von jeder beliebigen Pose in jede beliebige Pose gelangen [42]. Ein Beispiel dafür sind Autos, die sich in der Regel nicht omnidirektional bewegen können. Bei Autos ist die Anzahl der effektiven Freiheitsgrade (der Dimensionalität der Pose) größer als die der aktiven Freiheitsgrade (der translatorischen und rotatorischen Bewegungsmöglichkeiten des Fahrzeugs). Wenn Bewegungen für ein solches System geplant werden sollen, müssen seine nicht-holonomen Einschränkungen berücksichtigt werden. Erste Verfahren zur Planung solcher Systeme wurden von Dubins [30] für ein ausschließlich vorwärts fahrendes und später von Reeds und Shepp [80] für ein vorwärts und rückwärts fahrendes Auto vorgestellt. Die geplanten Wege werden in beiden Ansätzen aus Linien und Kreissegmenten zusammengesetzt und haben daher einen diskontinuierlichen Krümmungsverlauf [34]. Eine Erweiterung bietet der Continous Curvature Steer [34] Algorithmus, in dem er Klothoiden zur Verbindung von Linien und Kreissegmenten nutzt. Dadurch liefert er im Gegensatz zu [30] und [80] Pfade mit kontinuierlichem Krümmungsverlauf. Um Pfade nachträglich zu glätten, existieren Ansätze der Pfadglättung. Diese können zur Beseitigung von Diskontinuitäten des Krümmungsverlaufs eingesetzt werden. Zur Glättung eignen sich neben Klothoiden [60] beispielsweise auch B-Splines [52], quintische Polynome [97], polare Splines [69] und kubische Spiralen [46]. Ein anderer Ansatz, um abfahrbare Trajektorien zu erzeugen, ist der Einsatz der Theorie der optimalen Steuerung [18]. Ein Verfahren zur reaktiven Trajektorienplanung nicht-holonomer Systeme wird beispielsweise von Kelly und Nagy [51] beschrieben. Ein neuerer Ansatz ist die Planung in einem Zustandsgitter mit einem Regelungsset [75]. Das Regelungsset gibt mögliche Bewegungen vor, die zur Bewegung von einer Zelle des Gitters zu einer anderen genutzt werden können. In dem Zustandsgitter wird die Planung durchgeführt. Dieser Ansatz wird beispielsweise von der Search-Based Planning Library (SBPL) [58], einer aktuellen Bibliothek zur Bewegungsplanung, genutzt.

### 3.1.4 Planung für kooperierende mobile Roboter

Eine grundlegende Aufgabenstellung für kooperierende mobile Roboter ist die Planung von Bewegungen, die zu der Erfüllung eines gemeinsamen Ziels führen, während gegenseitige Behinderungen und Kollisionen vermieden werden [74]. Diese Aufgabenstellung muss beispielsweise für Fahrmanöver kognitiver Automobile gelöst werden [36]. In [66, 91] wird der kooperative dreidimensionale Transport von Nutzlast mit mehreren Quadrocoptern beschrieben.

Multi-Roboter Systeme ermöglichen unter anderem das Erfüllen von Aufgaben, die ein einzelner Roboter alleine nicht stemmen kann. Ein Beispiel dafür ist als Instanz des Problems der Museumswächter [73] der Einsatz von mobilen Robotern für kooperative Überwachung. Die Planung für kooperierende mobile Roboter ist seit Jahrzehnten ein aktives Forschungsgebiet, für das es eine Vielzahl an Anwendungen gibt. In [21] werden beispielsweise Arbeiten aus einigen Anwendungsfeldern der Planung für kooperierende mobile Roboter beschrieben. Diese umfassen die

Verkehrssteuerung [38], die kooperative mobile Manipulation [64, 84, 89, 99] und die Futtersuche [5, 10, 29, 37, 92, 95], die Ernteszenarien einschließt.

Die Verfahren zur Planung für kooperierende mobile Roboter können in zentrale und dezentrale Ansätzen klassifiziert werden [102]. Bei ersteren werden die Pläne der Fahrzeuge in einem zentralen Planungsschritt für alle beteiligten Roboter berechnet. Bei dezentralen Ansätzen werden beispielsweise Verfahren aus dem Forschungsbereich der Multi-Agenten-Systeme (MAS) aus der Künstlichen Intelligenz eingesetzt. Die Forschung auf diesem Gebiet liefert Methoden zur Koordination und Kooperation von autonom handelnden Agenten [94, 102].

Im Folgenden werden Kooperationsmechanismen aus der Robotik aufgelistet [36] und aktuelle Arbeiten genannt. Der biologisch inspirierte Kooperationsmechanismus der Schwarmrobotik hat zum Ziel, mit simpel handelnden Robotern ein komplexes Schwarmverhalten zu erreichen [28, 31]. Motivation ist beispielsweise das Nachbilden des Schwarmverhaltens von Fischen oder Vögeln durch die Interaktion benachbarter Roboter [31]. Der Ansatz der Ameisenkolonie Optimierung nutzt gelegte Duftspuren zur Interaktion und dient dem Auffinden eines kürzestes Wegs [28]. Aktuelle Arbeiten der Schwarmrobotik werden in [17] beschrieben. Ein weiterer Kooperationsmechanismus ist die kooperative Regelung. Diese wird beispielsweise für Formationsfahrten von Robotern genutzt, bei der die Position von Formationsmitgliedern häufig relativ zu einem Führungsfahrzeug geregelt wird [33, 36, 70]. Die Erstellung eines Graphen aus der Umgebung und eine zentrale koordinierte Bewegungsplanung mobiler Roboter in diesem Graphen ist ein weiterer Ansatz [96].

An der Vielfalt der Ansätze wird deutlich, dass es keinen allgemeingültigen Ansatz zur Planung für kooperierende mobile Roboter gibt. Je nach Anwendungsfall und Randbedingungen muss ein geeignetes Verfahren ausgewählt werden. Für eine prozessoptimierte Planung in dem Infield-Transportlogistik Szenario bietet sich ein zentraler Ansatz an, der mit globaler Sicht den Gesamtprozess optimieren kann.

### 3.1.5 Prozessoptimierung

Für eine Optimierung kommen Ansätze verschiedener Forschungsbereiche in Frage. Der Bereich des Operations Research (OR) liefert Algorithmen für praktische Planungsprobleme, die auf mathematische Optimierungsverfahren [27] zurückgreifen. Der Bereich der Computational Intellgence (CI) [31] hingegen nutzt vor allem biologisch inspirierte Verfahren für die Optimierung. Ein Beispiel dafür sind evolutionäre Algorithmen [103], bei denen durch Selektions-, Rekombinations und Mutationsschritte über mehrere Generationen (Iterationen) immer bessere Indivduen (Lösungen) generiert werden. Ein weiteres Beispiel der CI ist die Schwarmintelligenz [12], zu der die Partikelschwarm [76] und die Ameisenkolonie Optimierung [28] gehören.

Dass es kein allgemeingültiges bestes Verfahren zur Optimierung für alle Problemstellungen gibt, wird in [105] mit dem No-Free-Lunch-Theorem für die Optimierung dargelegt.

### 3.1.6 Robotik in der Landwirtschaft

Der Einsatz von Robotern und robotischer Technologien und Verfahren in der Landwirtschaft ist ein logischer Schritt zur Autonomie- und Effizienzsteigerung. In dem folgenden Abschnitt werden zwei Forschungsprojekte vorgestellt, die den Stand der Technik von mobilen robotischen Systemen in der Landwirtschaft widerspiegeln. Anschließend wird der Stand der Technik in der Planung für landwirtschaftliche Roboter vorgestellt.

In BoniRob [82] wurde ein autonomer Feldroboter entwickelt, der durch Bildverarbeitung und GPS-Sensorik Informationen zu Einzelpflanzen auf einem Feld ermitteln und hinterlegen kann. Bei diesem Projekt wurde eine neue robotische Plattform entwickelt, in die landwirtschaftliche Funktionalität integriert wurde. Ein anderer Ansatz zum Technologietransfer wird in dem aktuellen Forschungsprojekt AgroBot [101] verfolgt. Hier werden robotische Technologien und Verfahren in eine bestehende Landmaschine integriert. In AgroBot wird ein Rübenroder mit Sensorik und einem Planungssystem versehen, so dass er autonom einen Ernteprozess durchführen kann. Besonderer Fokus liegt bei der Planung auf Bodenschonung durch Minimierung von Fahrstrecken und Mehrfachüberrollungen. Beide Ansätze bieten ihre eigenen Vorteile. Der Aufbau neuer robotischer Systeme bietet eine hohe Flexibilität, beispielsweise den Einsatz von Schwärmen autonomer mobiler Roboter. Die Integration von robotischen Technologien und Verfahren in bestehende und am Markt platzierte Systeme kombiniert den Stand der Technik des Maschinenbaus in der Landtechnik mit einem schnellen Technologietransfer in kommerziell verfügbare Produkte.

Für die Problemstellung der Bestimmung des optimalen, vollständig abdeckenden Wegs eines Mähers mit Wendekosten wird in [4] die NP-Vollständigkeit bewiesen und ein polynomieller Approximationsalgorithmus vorgeschlagen. In [8] wird der Einsatz der Ameisenkolonie Optimierung für die abdeckende Pfadplanung in einem topologischen Graphen für eine einzelne Arbeitsmaschine mit verschiedenen Wendestrategien beschrieben. Oksanen stellt in [71] zwei weitere Verfahren zur abdeckenden Pfadplanung für Landmaschinen durch Dekomposition vor. In [39, 90] werden Verfahren zur Planung optimierter und flächendeckender Abarbeitung durch eine Arbeitsmaschine für Felder mit Hindernissen und unter Berücksichtigung von Wendekosten vorgestellt. Diese Verfahren sind jeweils auf einzelne Arbeitsmaschinen ausgelegt. In dem Infield-Transportlogistik Szenario, das in dieser Arbeit betrachtet wird, sind zusätzlich Transportfahrzeuge zur Unterstützung von Arbeitsmaschinen zu berücksichtigen.

Ein hierarchischer Ansatz zur generellen Prozessplanung von Landmaschinen wird in [15] vorgeschlagen. Thematisiert werden einzelne, nachgeordnete Planungsschritte der Zuweisung von Arbeitsmaschinen zu Feldern und die Planung von Fahrrouten als Instanzen des Vehicle Routing Problems (VRP). In [13, 14] wird der Vorschlag konkretisiert, die Aufgabenstellung der Infield-Transportlogistik als VRP zu definieren, um aus der Forschung des OR und vorhandenen Algorithmen zu profitieren. Eine wesentliche Einschränkung ist, dass für diese Formulierung die Service Orte (Kunden in der Terminologie des VRP) a priori bekannt sein müssen. Das Kreuzen von Fahrspuren wird von dem Ansatz ebenfalls nicht ermöglicht, sondern zum Fahrspurwechsel muss das Transportfahrzeug immer das Vorgewende nutzen. Ein Verfahren zur Planung eines Infield Servicefahrzeugs, das Arbeitsmaschinen unterstützt, wird in [45] beschrieben. Das Ver-

fahren nutzt eine topologische zweidimensionale Rasterdarstellung des Felds zur Planung. Das Kreuzen von Fahrspuren wird auch hier nicht berücksichtigt und eine Fahrspur ist immer entweder vollständig abgeerntet oder frei. Die Diskretisierung der abgeernteten Fläche ist dadurch sehr grob. Die Infield-Transportlogistik der Getreideernte wird in [3] als Minimum Cost Network Flow Problem (MCNFP) [35] formuliert. Dazu wird das Feld in einer Rasterkarte diskretisiert. Der Ansatz eignet sich allerdings nur für sehr kleine Felder [3] und sieht keine Planung von Parallelfahrten zum Überladen vor.

Keines der vorgestellten oder in der Literatur gefundenen Verfahren bietet eine Lösung für die in Abschnitt 2.3 beschriebene Aufgabenstellung der Planung des Infield-Transportlogistik Szenarios.

## 3.2 Planung in *marion*

In diesem Abschnitt werden die technischen Randbedingungen (Abschnitt 3.2.1) und funktionalen Anforderungen (Abschnitt 3.2.2) des Projekts *marion* beschrieben. Anschließend werden die Architektur (Abschnitt 3.2.3) sowie der genutzte Framework und die Kommunikationsstruktur (Abschnitt 3.2.4) des Planungssystems vorgestellt.

### 3.2.1 Technische Randbedingungen

#### Kommunikation

Damit die Maschinen ihre Pläne aufeinander abstimmen können, ist eine Kommunikation zwischen den Maschinen erforderlich. In den ländlichen Gebieten, in denen Getreide angebaut wird, ist zu der Projektlaufzeit nicht mit flächendeckend verfügbarem Mobilfunk zu rechnen. Daher muss eine andere Funktechnologie genutzt werden. Die technische Umsetzung der Feldkommunikation liegt in dem Projekt in der Verantwortung des Partners CLAAS. Es ist grundsätzlich davon auszugehen, dass die Bandbreite der Funkkommunikation begrenzt ist, schwankt und während der Prozessausführung abreißen kann. Daher ist eine Anforderung an das Planungssystem, möglichst wenig Bandbreite zu benötigen und auf Kommunikationsaussetzer reagieren zu können.

#### Sensorik

Mobile Roboter besitzen in der Regel Sensoren, mit denen sie ihr Umfeld wahrnehmen können, zum Beispiel zur Hindernisvermeidung (siehe Abschnitt 3.1). In dem Projekt *marion* verfügen die Feldfahrzeuge jedoch über keinerlei Sensorik zur Abstandsmessung. Es ist auch nicht geplant, in der Projektlaufzeit Hindernissensorik auf die Maschinen und in das Planungssystem zu integrieren. Das hat zum einen den Grund, dass eine Unterscheidung zwischen dem gewollten Fahren des Mähdreschers in den Bestand beim Ernten und eine ungewollte Kollision mit statischen Hindernissen beim Ernten nicht leicht zu treffen ist. Zum anderen werden aus rechtlichen

und prozesstechnischen Gründen weiterhin Fahrer auf dem Fahrersitz der Maschinen sitzen. Diese Fahrer haben unter anderem die Aufgabe der Hindernisvermeidung, die bei vollautonomen Systemen sonst durch die Systeme selbst erfüllt werden muss. Für die Positionsbestimmung verfügen die Feldfahrzeuge über *Global Positioning System* (GPS) Antennen. Durch eine zusätzliche stationäre Referenzstation, die sich hochgenau einmisst und ein Korrektursignal sendet, können sich die Maschinen auf wenige Zentimeter genau lokalisieren. Die Position wird über den *Controller Area Network* (CAN) Datenbus der Maschinen als Längen- und Breitengradangabe in *World Geodetic System 1984* (WGS84) Geokoordinaten zur Verfügung gestellt. Der Füllstand des Mähdreschers wird anhand des in der Maschine gemessenen Durchsatzes und Ertrags berechnet und ebenfalls per CAN bereitgestellt. Für den Mähdrescher sind dort außerdem noch der Zustand der Korntankentleerschnecke und die Position des Korntankrohrs verfügbar.

#### Lenksysteme

Die Performanz der Lenksysteme ist a priori nicht bekannt und muss im Rahmen der Arbeit analysiert werden, um abfahrbare Trajektorien für die realen Maschinen generieren zu können.

#### Prozessdynamik

Der Prozess der Getreideernte unterliegt naturgemäß einer hohen Dynamik. Vor allem schwankende Erträge können zu einer vom Plan abweichenden Prozessausführung durch variierende Geschwindigkeiten und Korntankfüllstände des Mähdreschers führen.

#### Software

Das Planungssystem soll auf Fahrzeugrechnern mit Windows 7 integriert werden und es sollen nur Bibliotheken verwendet werden, die frei für kommerzielle Nutzung einsetzbar sind.

### 3.2.2 Funktionale Anforderungen an das Planungssystem

Die funktionalen Anforderungen, die an das Planungssystem gestellt werden, basieren auf der Beschreibung der Planungsaufgabe aus Abschnitt 2.3 unter den in Abschnitt 3.2.1 beschriebenen technischen Randbedingungen.

Zusammenfassend soll das Planungssystem einen gültigen, gemeinsamen Plan für die Ernte und den Abtransport des Ernteguts durch die Feldfahrzeuge erstellen. Dieser maschinenübergreifende Plan muss auf einem Fahrzeugrechner vor Prozessbeginn in annehmbarer Zeit erstellt und bei Abweichungen in der Prozessausführung dynamisch angepasst werden können. Der gemeinsame Plan muss an alle Maschinen verteilt werden, damit die Maschinen in dem Prozess kooperieren können. Dabei müssen die Maschinen den Plan tatsächlich ausführen können. Daher sind kinematische und prozesstechnische Einschränkungen der Fahrzeuge einzuhalten. Das Planungssystem

soll möglichst wenig Kommunikationsbandbreite benötigen und auf Abrisse der Funkverbindung reagieren können.

### 3.2.3 Architektur des Planungssystems

Das Architekturkonzept des Planungssystems begründet sich unter anderem in den technischen Randbedingungen. Es wird für das Planungssystem eine hierarchische Aufteilung in die Schichten der übergeordneten Routenplanung, der fahrzeugspezifischen Bewegungsplanung und der maschinennahen Planausführung gewählt (siehe Abbildung 3.1).

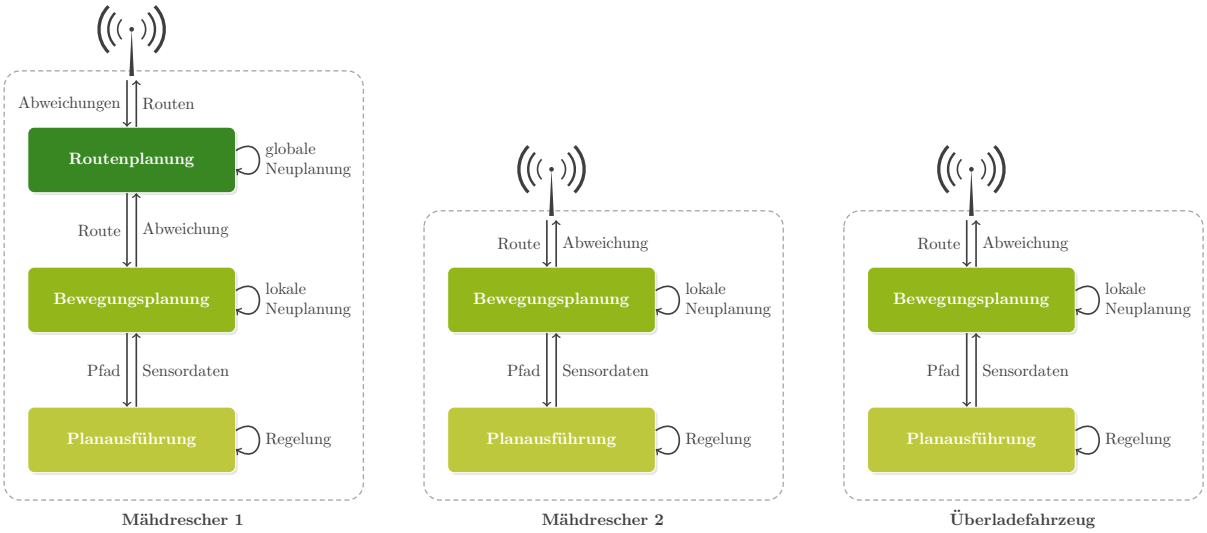



**Abbildung 3.1:** Architektur des Planungssystems (angepasst von [81]). Es ist in drei Schichten unterteilt. Die zentrale übergeordnete Routenplanung übernimmt die logistische Planung für alle Feldfahrzeuge und somit die Optimierung des Gesamtprozesses. Die Bewegungsplanung verfeinert mit fahrzeugtypspezifischen Verfahren die grobgranularen Routenpläne in feingranulare, durch die Maschinen ausführbare Pläne. Die maschinennahe Schicht der Planausführung übernimmt die Regelung der Maschinen auf die jeweiligen Pfade.

#### Routenplanung

Die übergeordnete Schicht der Routenplanung übernimmt die logistische Planung des Gesamtprozesses für alle beteiligten Feldfahrzeuge. Es wird für die Erntefahrzeuge der Ernteprozess und für die Erntefahrzeuge und das Überladefahrzeug der gemeinsame Abtransport des Ernteguts bis zum Feldrand geplant. Wie in Kapitel 2 beschrieben, beeinflussen sich diese Pläne für Ernte und Transport gegenseitig in Raum und Zeit. Es ist daher erforderlich, dass der maschinenübergreifende Plan konsistent ist. Also müssen auf allen Feldfahrzeugen Routenpläne vorliegen, die denselben Ablauf für den Gesamtprozess darstellen. In den Routenplänen müssen außerdem die kooperativen Prozessabschnitte beschrieben sein.

Der Routenplan für einen Mähdrescher enthält die Fahrspurreihenfolge, die den Ernteprozess beschreibt und die Abtankvorgänge, bei denen der Korntankinhalt über das ausgeschwenkte Korntankrohr in Parallelfahrt auf das Überladefahrzeug überladen wird. Für das Überladefahrzeug besteht der Routenplan aus einer Folge von Überladevorgängen, bei denen jeweils eine bestimmte Menge an Erntegut von einem Mähdrescher aufgenommen wird, und aus Abladevorgängen am Feldrand, bei denen der Inhalt des Überladewagens in den Straßentransporter entladen wird.

Die Routenplanung wird zentral auf einem vor Prozessbeginn festgelegten Fahrzeugrechner für alle beteiligten Feldfahrzeuge durchgeführt. Auf diesem Rechner müssen dafür alle prozessrelevanten Informationen als Planungsgrundlage zur Verfügung stehen. Die zentrale Planung stellt sicher, dass keine konkurrierenden Pläne existieren und dass die berechneten Routenpläne konsistent sind. Diese Pläne werden per Funk an die beteiligten Feldfahrzeuge verteilt. Um mit möglichst wenig Bandbreite auszukommen, werden die Routenpläne durch wenige markante Routenpunkte beschrieben. Die einzelnen Routenpunkte sind jeweils orts-, orientierungs-, geschwindigkeits-, zeit- und füllstandsbehaftet. Diese Attribute sind erforderlich, um die Kooperation der Maschinen präzise und eindeutig zu beschreiben. Für einen Mähdrescher besteht die Route aus zwei Routenpunkten für die Ein- und Ausfahrt jeder abzuarbeitenden Fahrspur, und aus zwei Routenpunkten für Start und Ende jeder Überladeprozedur. Die Gesamtzahl der Routenpunkte für einen Mähdrescher beträgt also $2n + 2u$ für $n$ abzuarbeitende Fahrspuren und $u$ Überladevorgänge. Für das Überladefahrzeug wird entsprechend jeder Überlade- und Abladevorgang durch einen Start- und einen Endpunkt repräsentiert. Die Anzahl an Routenpunkten für das Überladefahrzeug beträgt also $2u+2a$ für $u$ Überladeprozeduren in Parallelfahrt und $a$ Entladevorgänge am Feldrand. Die geringe benötigte Bandbreite für die Verteilung der Routenpläne ist ein wichtiger technischer Grund für die gewählte Architektur. Ein wesentlicher algorithmischer Grund ist die Verringerung der Planungskomplexität.

Die Routenplanung hat als übergeordnete Planungsschicht die globale Prozesssicht und die Aufgabe, den Gesamtprozess zu optimieren. Die nachgeordneten Schichten hingegen betrachten jeweils nur die fahrzeugspezifischen Teilprozesse. Durch diese Aufteilung kann in der Schicht der Routenplanung mit grober Granularität effizient in einem topologischen Graphen geplant werden. Die Bewegungsplanungsschicht jedes Fahrzeugs verfeinert diesen Plan anschließend. Müssten in jedem Optimierungsschritt des Gesamtprozesses durch die Routenplanung feingranulare, durch die Maschinen ausführbare Pläne generiert werden, so wäre aufgrund der Planungskomplexität keine praktikable Laufzeit auf den Fahrzeugrechnern erreichbar.

Die Planungsschicht der Routenplanung wird eingehend in Kapitel 4 beschrieben.

#### Bewegungsplanung

Aufgabe der Bewegungsplanung ist es, die Routenpläne so zu verfeinern, dass sie von den Maschinen ausgeführt werden können. Für die unterschiedlichen Fahrzeugtypen müssen aufgrund der abweichenden Vorgaben unterschiedliche Bewegungsplanungen realisiert werden. Ein Mähdrescher soll beim Ernten einer Fahrspur dieser möglichst genau folgen, während er beim Wenden

frei auf abgeernteter Fläche navigieren darf. Das Überladefahrzeug hingegen muss den Fahrspuren nicht immer genau folgen, sondern darf diese zusätzlich mit erhöhten Kosten kreuzen. Das Überladefahrzeug darf sich dabei aber im Gegensatz zu Erntefahrzeugen nur auf bereits abgeernteter Fläche bewegen.

Das Ergebnis der fahrzeugtypspezifischen Bewegungsplanung ist immer ein durch das jeweilige Fahrzeug abfahrbarer Pfad, der für die Planausführung bereitgestellt wird. Die Planungsschicht der Bewegungsplanung wird in Kapitel 5 detailliert dargestellt.

#### Planausführung

Die maschinennahe Planausführung wurde in dem Projekt *marion* durch den Projektpartner CLAAS realisiert. Die eingesetzten Lenksysteme sollen Wegpunkte der Bewegungsplanung entgegen nehmen und die Fahrzeuge mit hinreichender Genauigkeit auf diese Sollstrecken regeln können. Die Anbindung der Planausführung an die Bewegungsplanung ist in Abschnitt 5.1.1 beschrieben. Eine weitere wichtige Aufgabe der Planausführung ist die Bereitstellung von Sensordaten für die übergeordneten Schichten des Planungssystems.

#### Dynamische Neuplanung

Da bei der Prozessausführung mit Abweichungen zu rechnen ist, muss eine dynamische Neuplanung durch das Planungssystem realisiert werden. In dem gewählten Ansatz wird immer erst versucht, auf Abweichungen in der Prozessausführung lokal und auf möglichst tiefer Planungsebene zu reagieren. Für die Planausführungsschicht bedeutet das, dass bei Abweichungen von der Sollstrecke als erstes die Regelung durch das Lenksystem eingreift. Stellt die Schicht der Bewegungsplanung jedoch fest, dass die Abweichung einen Toleranzwert überschreitet, so wird ein neuer Pfad für die aktuelle Route des Fahrzeugs berechnet. Bis zu diesem Punkt betreffen die Planänderungen nur die Maschine, deren Ausführung vom Plan abweicht. Erst wenn die Bewegungsplanung die Route nicht mehr genau genug einhalten kann, schreitet die Routenplanung ein und erstellt einen neuen maschinenübergreifenden Routenplan für alle beteiligten Feldfahrzeuge. Eine Neuplanung durch die Routenplanung ist aber nur dann zulässig, wenn Funkkommunikation zwischen den Fahrzeugen verfügbar ist. Der beschriebene Ansatz der dynamischen Neuplanung sorgt dafür, dass die Funkkommunikation auf das Nötigste beschränkt wird.

### 3.2.4 Framework und Kommunikationsstruktur

Die Implementierung des Planungssystems geschieht in der Programmiersprache C++ unter Verwendung des plattformunabhängigen Programmierwerkzeugs CMake [43]. Es werden die folgenden Programme und Bibliotheken von Drittanbietern genutzt.

- **boost**: Weit verbreitete Sammlung an C++ Bibliotheken [43] mit hohen Qualitätsansprüchen

- **Search-Based Planning Library (SBPL)**: Bibliothek für die Bewegungsplanung nach aktuellem Stand der Technik [58]
- **Qt**: Bibliothek für plattformübergreifende Programmierung grafischer Benutzeroberflächen. [78]
- **PrismTech OpenSpliceDDS**: Datenzentrische Kommuninkationssoftware [77]

Das Planungssystem setzt sich modular zusammen aus Komponenten, die Funktionalitäten kapseln. Alle Komponenten werden als dynamische Bibliotheken erstellt. Zur Ausführung von einer oder mehreren Komponenten wird ein Startprogramm entwickelt. Die zu startenden Komponenten und deren Parameter werden dem Startprogramm in einer Konfigurationsdatei übergeben. Der Starter lädt alle dynamischen Bibliotheken des Planungssystems und startet die geforderten Komponenten mit den angegebenen Parametern.

Die Kommunikation der Komponenten untereinander wird nach dem datenzentrischen *Publisher-Subscriber* Konzept des *Data Distribution Service* (DDS) realisiert. DDS ist ein von der *Object Management Group* (OMG) festgelegter Standard [72].

Die zu versendenden Datentypen werden bei DDS in einer *Interface Definition Language* (IDL) Datei beschrieben. Diese Datei kann für verschiedene Programmiersprachen übersetzt werden. Aus der IDL Datei werden die C++ Dateien erstellt, die unter den Komponenten austauschbare Datentypen beschreiben. Die IDL Datei wird auch als Schnittstellenbeschreibung der durch das DFKI und durch CLAAS erstellten Komponenten genutzt. In der IDL Datei sind daher zusätzlich Kommentare zu DDS-spezifischen Parametern hinterlegt, beispielsweise zur Dienstgüte (*Quality of Service* (QoS)).

In dem Projekt *marion* wird die Implementierung *OpenSpliceDDS* der Firma *PrismTech* verwendet. Die Module des Planungssystems sind jeweils mit einer Klasse zur Anbindung an den DDS gekapselt, um durch Trennung von Funktion und Kommunikation den Transfer der Funktionalitäten in andere Frameworks zu erleichtern.

Posebezogene Informationen werden zwischen allen Komponenten immer in dem geodätischen WGS84 Koordinatensystem ausgetauscht. Intern rechnen die durch das DFKI erstellten Komponenten der oberen beiden Planungsschichten nicht in geodätischen, sondern in lokalen kontinuierlich metrischen kartesischen Koordinaten. Die Koordinaten werden daher zwischen dem ellipsoiden WGS84 und dem planaren metrischen *Universal Transverse Mercator* (UTM) Koordinatensystem für Planung und Kommunikation hin- und rückprojiziert.

Für die Integrations- und Erntetests auf den realen Maschinen werden Installationsprogramme für die Fahrzeugrechner erstellt. Durch diese Programme werden alle für die jeweiligen Tests erforderlichen Komponenten, Bibliotheken und Konfigurationsdateien auf dem Fahrzeugrechner hinterlegt und eingerichtet.

# Kapitel 4

# Routenplanung

In diesem Kapitel wird die für den Anwendungsfall der Infield-Transportlogistik entwickelte maschinenübergreifende Planung eingehend beschrieben. Diese übergeordnete Planung wird im Folgenden als Routenplanung bezeichnet. Für das Projekt *marion* wurde die Planung für den in Abschnitt 2.2 beschriebenen Anwendungsfall der Ernte und des Transport des Ernteguts bis zum Feldrand mit zwei kooperierenden Feldfahrzeugen realisiert. Die Routenplanung benötigt für die Planung Informationen zu den eingesetzten Fahrzeugen, dem zu bearbeitenden Feld und zu dem Ernteprozess. Aus diesen Informationen werden Routenpläne für die beteiligten Feldfahrzeuge erstellt (Abbildung 4.1). Die Routenpläne werden hinsichtlich des Gesamtprozesses optimiert und berücksichtigen daher gleichermaßen den Ernte- und den Transportprozess.

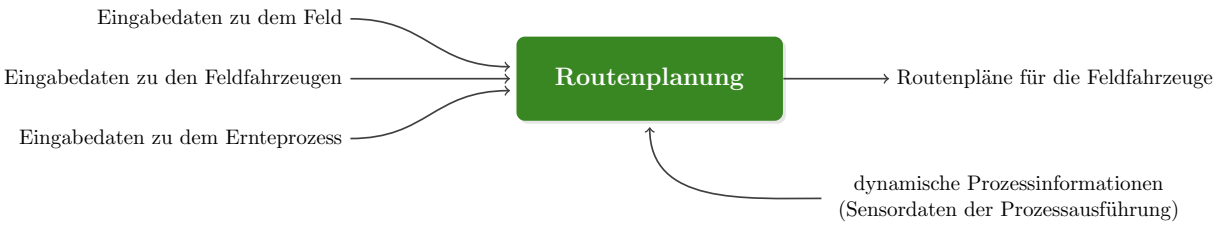



**Abbildung 4.1:** Informationsbedarf der Routenplanung. Für die Routenplanung werden initial Informationen zum Feld, zu den Feldfahrzeugen und zu dem Ernteprozess benötigt. Für die dynamische Plananpassung bei der Prozessausführung werden zusätzlich Sensordaten, wie beispielsweise die aktuellen Füllstande der Fahrzeuge, benötigt.

Die Datengrundlage und die Berechnungen der Eingabedaten für die Routenplanung wird in Abschnitt 4.1 beschrieben. Anschließend wird das Planungsproblem der Routenplanung in Abschnitt 4.2 in die zwei Planungsschritte der Berechnung der Fahrspurreihenfolge des Mähdreschers und der Überladeplanung unterteilt. Die Aufgabenstellung und das implementierte Planungsverfahren werden für die Fahrspurreihenfolge in Abschnitt 4.3 und für die Überladeplanung in Abschnitt 4.4 detailliert beschrieben. Abschnitt 4.5 fasst die Ergebnisse der Routenplanung zusammen, die in Abschnitt 4.6 abschließend diskutiert werden.

## 4.1 Datengrundlage der Routenplanung

Für die Routenplanung werden die in Tabelle 4.1 aufgelisteten feld-, fahrzeug- und prozessspezifischen Eingabedaten benötigt. Diese können teilweise aus Dateien eingelesen werden, beispielsweise die Feldumrandung oder die Korntankkapazität des eingesetzten Mähdreschers. Zum Teil müssen diese aber auch für den aktuellen Prozess aufbereitet werden. Ein Beispiel dafür sind die Fahrspuren, die aus der Feldgeometrie, der Arbeitsbreite des Mähdreschers und einer Referenzlinie berechnet werden müssen. Für die Bereitstellung und Berechnung von Eingabedaten für die Routenplanung existieren in der modularen Architektur des Planungssystems verschiedene Komponenten. In Abbildung 4.2 ist eine Übersicht der Komponenten gegeben, die Daten für die Routenplanung liefern. Die Kommunikation zwischen den Komponenten geschieht dabei per *Data Distribution Service* (DDS), wie in Abschnitt 3.2.4 beschrieben.

**Tabelle 4.1:** Eingabedaten der Routenplanung. Zur Planung werden feld-, fahrzeug- und prozessspezifische Eingabedaten benötigt.

| **Eingabe** | **Bezug**[1] | **Beschreibung** |
|---|---|---|
| Arbeitsbreite | MD | Breite des Schneidwerks, die effektiv zum Ernten genutzt wird |
| Ernteleistung | MD | maximale Ernteleistung des Mähdreschers |
| Korntank Kapazität | MD, ÜLF | fahrzeugspezifische maximale Ladungskapazität |
| Abtankleistung | MD, ÜLF | fahrzeugspezifische Leistung, mit der überladen wird |
| Fahrspur Fahrgeschwindigkeit | MD, ÜLF | fahrzeugspezifische Maximalgeschwindigkeit auf der Fahrspur |
| Vorgewende Fahrgeschwindigkeit | MD, ÜLF | fahrzeugspezifische Maximalgeschwindigkeit für Wendemanöver |
| Feldumrandungen | Feld | äußere und innere Feldumrandung |
| Fahrspuren[2] | Feld | Fahrspuren, die die innere Feldfläche für die gegebene Arbeitsbreite abdecken |
| Abtankort am Feldrand | Prozess | für den Ernteprozess festgelegter Ort am Feldrand, an dem das Überladefahrzeug die Ladung des Überladewagens in einen Straßentransporter entladen kann |
| maximaler Füllstand | Prozess | angepeilter maximaler prozentualer Füllstand für den Mähdrescher |
| Ertragsschätzung | Prozess | geschätzter Ertrag für das Feld zum Zeitpunkt der Prozessausführung |

[1] Abkürzungen: MD = Mähdrescher, ÜLF = Überladefahrzeug

[2] müssen vor der Planung aus Eingabedaten berechnet werden (siehe Abschnitt 4.1.4)

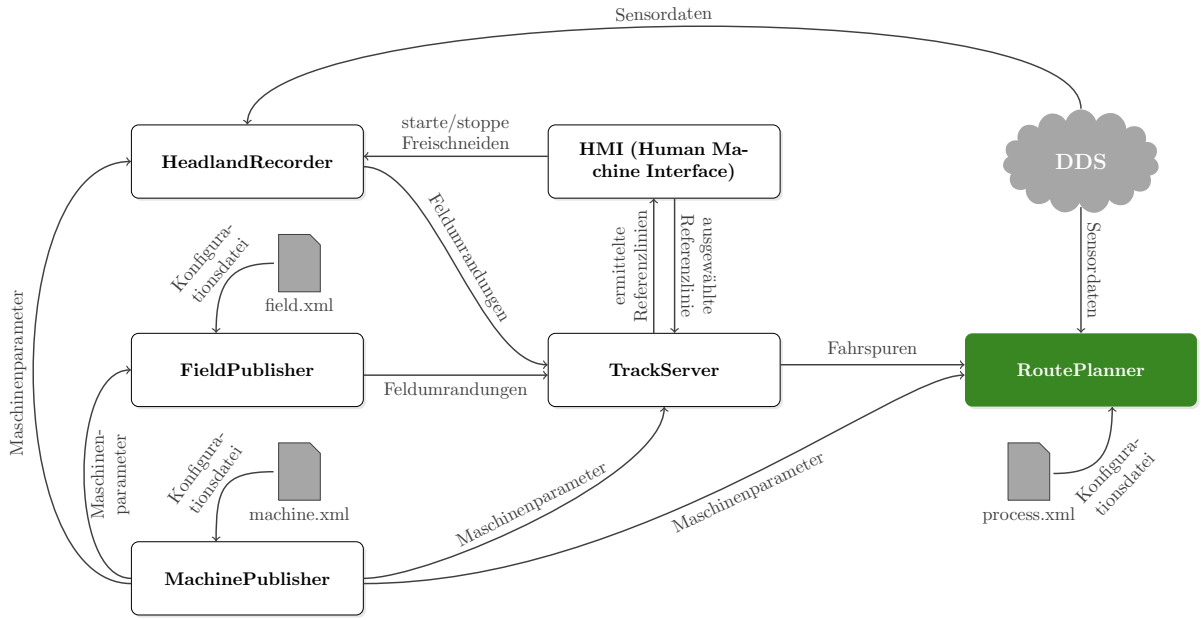



**Abbildung 4.2:** Komponenten zur Bereitstellung und Berechnung von Eingabedaten der Routenplanung.

Die fahrzeugspezifischen Eingabedaten werden auf den jeweiligen Maschinen durch die Komponente *MachinePublisher* aus Konfigurationsdateien ausgelesen und den Komponenten des Planungssystems bereitgestellt. Für die Fahrspurberechnung werden unter anderem die innere und äußere Feldumrandung benötigt. Diese Feldumrandungen können entweder durch den *FieldPublisher* aus einer Datei eingelesen werden, oder beim Freischneiden des Vorgewendes durch den *HeadlandRecorder* aufgezeichnet werden, wie in Abschnitt 4.1.2 beschrieben. Die Komponente *TrackServer* analysiert die Feldgeometrie und ermittelt nach dem in Abschnitt 4.1.3 beschriebenen Vorgehen mögliche Referenzlinien. Als Referenzlinie wird ein aus der Feldumrandung extrahierter Linienzug bezeichnet, der die Grundlage der Fahrspurberechnung bildet. Kandidaten für die Referenzlinie werden dem Fahrer des Mähdreschers vorgeschlagen, auf dem die Routenplanung ausgeführt wird. Nach der Auswahl der Referenzlinie liegen dem *TrackServer* alle nötigen Informationen vor, um die Fahrspuren für die verbleibende Erntefläche geometrisch zu berechnen. Die Fahrspurberechnung wird in Abschnitt 4.1.4 ausführlich beschrieben. Schließlich werden die prozesspezifischen Eingabedaten von der Komponente *RoutePlanner*, die die Routenplanung durchführt, aus einer Konfigurationsdatei eingelesen. Mit Erhalt der aktuellen Posen und Füllstände der Feldfahrzeuge wird der Planungsprozess angestoßen. Die folgenden Unterkapitel beschreiben die nötigen Berechnungen zur Bereitstellung der Eingabedaten.

### 4.1.1 Geometrische Berechnung der inneren Feldfläche

Falls die Feldgeometrie schon aufgezeichnet und persistent abgespeichert wurde, oder falls der Prozess vor der Ausführung testweise geplant werden soll, kann die Feldgeometrie aus einer Datei

eingelesen und zur Verfügung gestellt werden. Liegt in dieser Datei nur die äußere Feldumrandung vor, so muss das Vorgewende für die Bestimmung der inneren Feldumrandung geometrisch berechnet werden. Dazu werden im Uhrzeigersinn alle Segmente der äußeren Feldumrandung wie in Abbildung 4.3 orthogonal nach innen um die voreingestellte Vorgewendebreite verschoben. Die Flächen, die zwischen der äußeren Feldumrandung und den verschobenen Segmenten liegen, werden sukzessiv zu einer Fläche vereint. Schneiden sich die Flächen zwei aufeinander folgender Segmente nicht, so wird die Fläche vor der Vereinigung um einen Kreis mit dem Radius der Vorgewendebreite um den gemeinsamen Punkt erweitert. In dem Beispiel aus Abbildung 4.3 ist dies bei den Punkten $D$ und $I$ der Fall. Die innere Feldumrandung ist der innere Ring der berechneten Fläche, die Vorgewendefläche ist die Fläche zwischen der inneren und äußeren Feldumrandung.

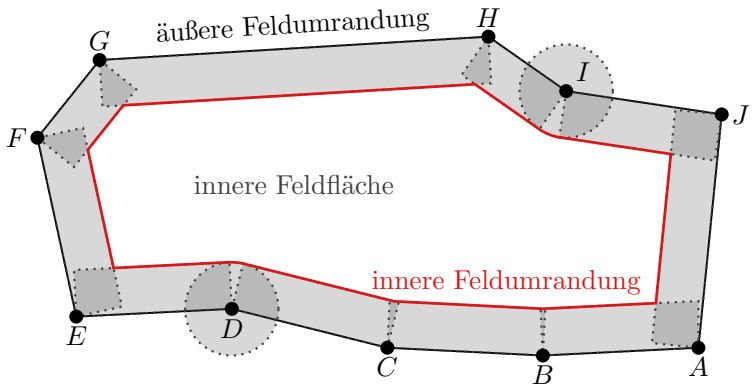



**Abbildung 4.3:** Geometrische Berechnung der inneren Feldfläche aus der äußeren Feldumrandung und einer voreingestellten Vorgewendebreite. Alle Segmente der äußeren Feldumrandung werden im Uhrzeigersinn orthogonal nach innen verschoben. Die Flächen zwischen ursprünglichen und verschobenen Segmenten werden vereinigt. Dabei werden Kreise eingefügt, falls sich Flächen nicht überschneiden, so wie in den Punkten $D$ und $I$. Der innere Ring der berechneten Fläche ist die innere Feldumrandung.

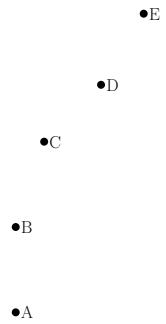
### 4.1.2 Aufzeichnung der Feldgeometrie während des Freischneidens

Bei der realen Prozessausführung in Erntetests auf dem Feld lagen vorab keine Feldumrandungen vor. Außerdem werden für die präzise Planung nicht die rechnerischen, sondern die real vorliegenden Feldumrandungen benötigt. Diese Feldumrandungen können aus der aufgezeichneten Vorgewendefläche extrahiert werden. Für die Bestimmung der freigeschnittenen Vorgewendefläche wurde eine Komponente des Planungssystems entwickelt, die die vom Mähdrescher befahrene Fläche während des Freischneidens berechnet. Die Komponente erhält die Maschinenpositionen des Mähdreschers beim Freischneiden in dem WGS84 Koordinatensystem mit einer Frequenz von 10 Hz per DDS. Aus den Positionen und anhand der Fahrzeuggeometrie, insbesondere der Arbeitsbreite, berechnet die Komponente kontinuierlich geometrisch die tatsächlich freigeschnittene Fläche. Dafür werden die geodätischen Positionsdaten in ein kartesisches metrisches Koordinatensystem überführt. Die polygonale Vorgewendefläche wird durch einen äußeren Ring und eine variable Anzahl an inneren Ringen geometrisch beschrieben. Unter den Annahmen, dass

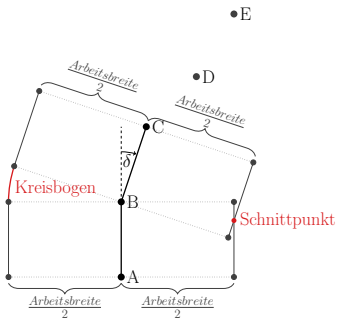
der größte innere Ring die innere Feldfläche repräsentiert, falls seine Fläche eine parametrierbare Mindestgröße aufweist und dass zusätzlich der äußere Ring immer die äußere Feldgrenze beschreibt, können die Feldumrandungen aus dem Vorgewendepolygon extrahiert werden.

Abbildung 4.4 zeigt den schematischen Ablauf der Konstruktion des Polygons, das die Vorgewendefläche repräsentiert. Nach dem Start der Aufzeichnung werden immer die aktuell erhaltene und die letzten zwei gültigen aufgezeichneten Positionen des Mähdreschers berücksichtigt. Hier wird zwischen den Punkten eine Mindestdistanz vorausgesetzt, um das Rauschen der hochfrequenten Positionsdaten zu verringern. Die zwei betrachteten Segmente, die sich aus den drei betracheteten Punkten ergeben, werden auf deren Winkeldifferenz $\delta$ untersucht. Überschreitet der Betrag der Wikeldifferenz einen Schwellwert $|\delta| \geq \frac{\pi}{2}$, so wird die Position verworfen. Auf diese Weise wird die Aufzeichnung von Rangiermanövern und Fehlern beim Erstellen des Vorgewendepolygons verhindert. Sind Winkeldifferenz und Abstand gültig, so werden die beiden Segmente um die halbe Arbeitsbreite orthogonal nach rechts und nach links verschoben. Schneiden sich die jeweils zur selben Seite verschobenen Segmente, wird ein Linienzug aus dem ersten Punkt des ersten verschobenen Segments, dem Schnittpunkt und dem zweiten Punkt des zweiten Segments erzeugt (Abbildung 4.4(b)). Schneiden sich die Segmente nicht, werden einem Linienzug der erste Punkte des ersten Segments, dann ein Kreisbogen ausgehend von dem Endpunkt des ersten Segments mit dem Radius der halben Arbeitsbreite bis zum ersten Punkt des zweiten Segments und schließlich der Endpunkte des zweiten Segments hinzugefügt (Abbildung 4.4(b)). Die Linienzüge, die aus den verschobenen Segmenten erzeugt wurden, werden zu einem geschlossenen Ring zusammengefasst (Abbildung 4.4(c)). Dieser Ring $R$ wird jeweils mit dem bisherigen Vorgewendepolygon $P$ geometrisch vereinigt $P = P \cup R$. So wächst das Polygon $P$ mit jeder neu erhaltenen und gültigen Position. Das Vorgewendepolygon wird anderen Komponenten und der Benutzerschnittstelle transformiert in WGS84 Geokoordinaten und mit einer Frequenz von 1 Hz über DDS bereitgestellt.

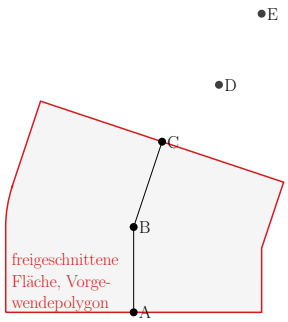
Sobald das Feld umrundet ist, entsteht der innere Ring, der die innere Feldfläche repräsentiert. Weitere innere Ringe entstehen, wenn beim Freischneiden nicht überlappend gefahren wird. Dies kann dadurch zustande kommen, dass ein Hindernis, wie zum Beispiel ein Strommast, umfahren werden muss. Diese kleineren inneren Ringe werden als Hindernisse im Vorgewende angesehen und gespeichert. Wie erläutert, wird der größte innere Ring als die innere Feldfläche angenommen, falls dessen Fläche eine voreingestellte Mindestgröße aufweist. Die innere Feldfläche beschreibt die Fläche, für die das Planungssystem einen flächendeckenden, möglichst optimalen, maschinenübergreifenden Plan erstellen soll. Wenn der Freischneidevorgang abgeschlossen ist, meldet der Fahrer dies durch eine Eingabe, woraufhin die Feldumrandungen persistent gespeichert und den Planungskomponenten per DDS zur Verfügung gestellt werden. In Abbildung 4.5 ist exemplarisch das Ergebnis der Aufzeichnung für zwei Freischneidevorgänge bei Erntetests gegeben.



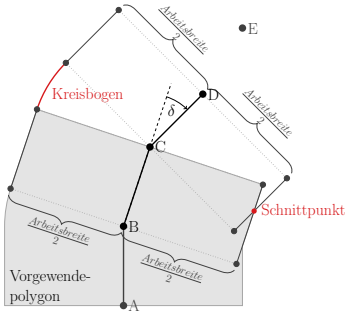

(a) Beispielpositionen A-E.



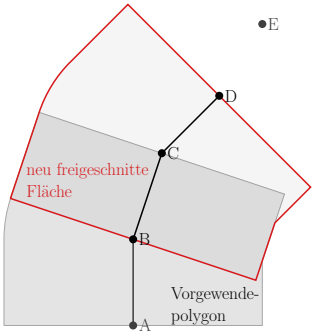

(b) Es wird die abgedeckte Fläche der jeweils letzten drei gültigen Positionen (hier: A-C) berechnet.



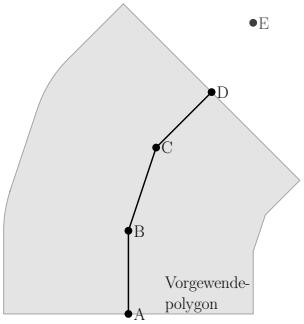

(c) Freigeschnittene Fläche für die Postionen A-C. Entspricht der Vorgewendefläche, da A-C die ersten Positionen waren.



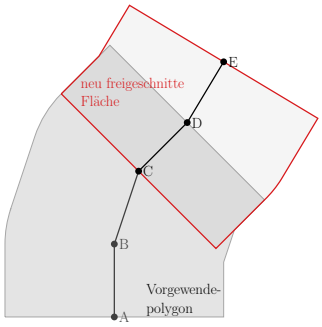

(d) Berechnung der abgedeckten Fläche für die Positionen B-D.



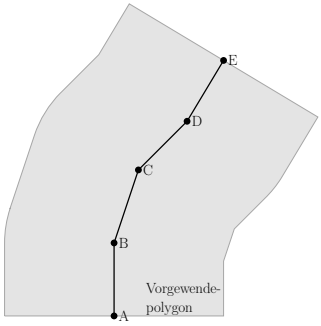

(e) Durch Positionen B-D abgedeckte und bislang freigeschnittene Vorgewendefläche.




(f) Neue Vorgewendefläche als Vereinigung der Flächen aus (e).

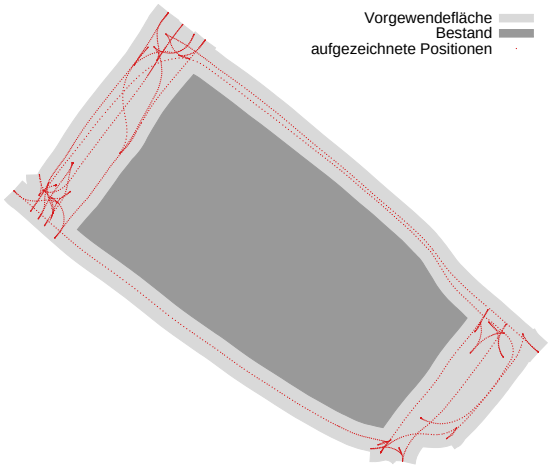



(g) Durch die Positionen C-E abgedeckte und bislang freigeschnittene Vorgewendefläche.

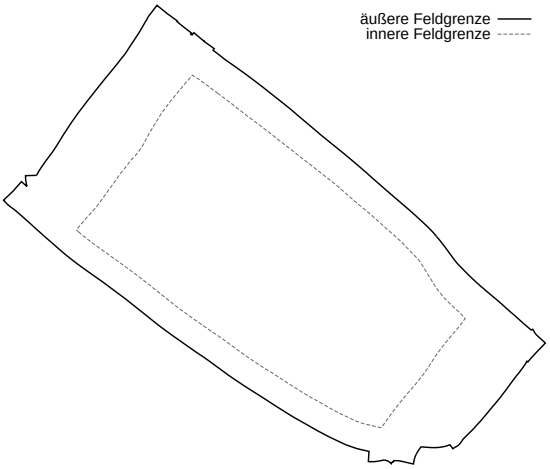



(h) Finale Vorgewendefläche für Positionen A-E.

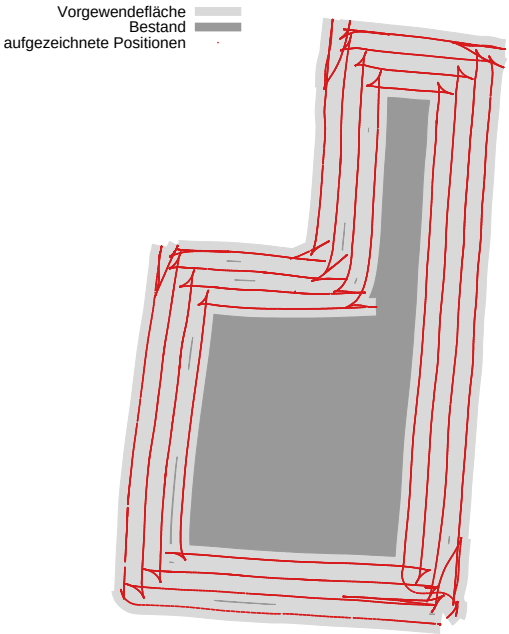
**Abbildung 4.4:** Exemplarische Darstellung des Aufzeichnens der freigeschnittenen polygonalen Vorgewendefläche für die Sequenz von fünf Positionen A-E aus (a). Beginnend mit der dritten gültigen Position (hier C) wird für die jeweils drei letzten Positionen anhand der Arbeitsbreite die freigeschnittene Fläche berechnet und sukzessiv mit der bislang freigeschnittenen Fläche geometrisch vereint. Das Ergebnis der Berechnung für die Positionen A-E ist in (h) gegeben.



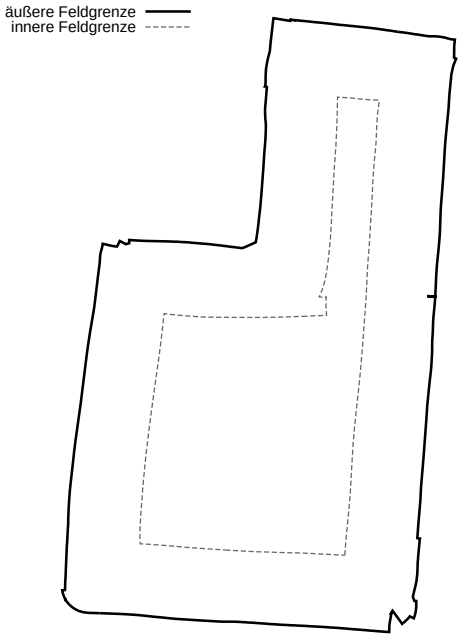

(a) Beispiel 1: Aufzeichnung des Freischneidevorgangs und die berechneten Flächen.




(b) Beispiel 1: Aus der aufgezeichneten Vorgewendefläche extrahierte Feldgrenzen.




(c) Beispiel 2: Aufzeichnung des Freischneidevorgangs und die berechneten Flächen.




(d) Beispiel 2: Aus der aufgezeichneten Vorgewendefläche extrahierte Feldgrenzen.

**Abbildung 4.5:** Beispielergebnisse für die Aufzeichnung des Vorgewendes mit unterschiedlichen Freischneidestrategien. Aus den aufgezeichneten Positionen und der Arbeitsbreite des Mähdreschers wird die freigeschnittene Fläche berechnet ((a) und (c)). Aus dem Ergebnis werden die äußere und innere Feldumrandung für die weitere Planung extrahiert ((b) und (d)).

### 4.1.3 Erkennung und Bewertung von Referenzlinien

Für die Berechnung der Fahrspuren muss eine Referenzlinie erzeugt werden. Die Referenzlinie ist ein Linienzug, der die Bearbeitungsrichtung des Feldes vorgibt. In der Praxis wird eine Kante der Feldumrandung als Referenzlinie benutzt. Die Fahrspuren ergeben sich aus der Parallelverschiebung dieser Referenzfahrspur. Die Fahrspurberechnung wird in Abschnitt 4.1.4 beschrieben.

Für die Referenzlinienerkennung wird die innere Feldgrenze geglättet, um lokale Ecken herauszufiltern. Die geglättete innere Feldgrenze wird dann auf verbliebene Ecken untersucht. Dafür wird an jedem Punkt der Kontur die Winkeldifferenz $\delta$ zwischen den angrenzenden Segmenten bestimmt. Überschreitet $\delta$ den Schwellwert $\frac{\pi}{7}$, so wird der Punkt als Eckpunkt in einer Liste gespeichert. Die Linienzüge zwischen den Eckpunkten stellen potentielle Referenzlinien dar (siehe Abbildung 4.6). Die mit dem gewählten Schwellwert für eine betrachtete Auswahl an realen Feldern vorgeschlagenen Referenzlinien wurden von Testkunden der Firma CLAAS und Experten der Landtechnik als plausible Vorschläge bestätigt.

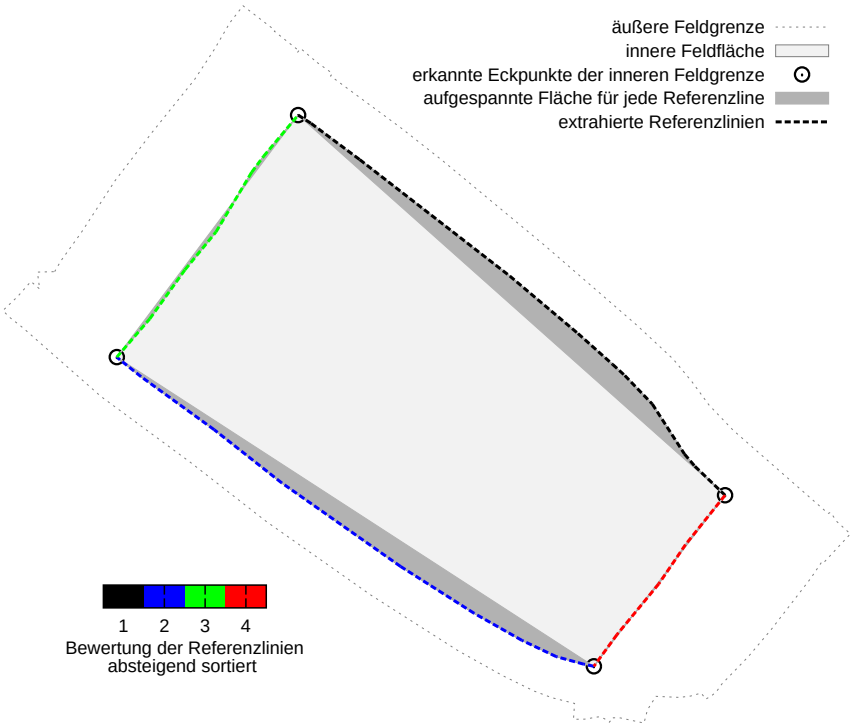



**Abbildung 4.6:** Erkennung von Referenzlinien anhand der inneren Feldumrandung am Beispiel des aufgezeichneten Feldes aus Abbildung 4.5(b). Ausgehend von Ecken, die in der Feldkontur erkannt wurden, werden Linienzüge aus der Feldumrandung extrahiert. Diese werden durch die Bewertungsfunktion aus Gleichung 4.1 bewertet und absteigend sortiert in einer Liste gespeichert. Die Ergebnisse der Bewertung für die dargestellten Referenzlinien sind in Tabelle 4.2 gegeben.

Die ermittelten Kandidaten der Referenzlinie werden hinsichtlich ihrer Verwendbarkeit bewertet, um ungültige Vorschläge zu vermeiden und um dem Benutzer die geeignetste Referenzlinie vorzuschlagen. Lange und gerade Feldkanten sind besonders gut als Referenzlinien geeignet. Des-

halb werden für alle möglichen Referenzlinien die Länge und die Form betrachtet. Um gerade Referenzlinien zu bevorzugen, wird für jeden Linienzug die geschlossene Fläche berechnet, die entsteht, wenn man seinen letzten Punkt mit seinem ersten Punkt verbindet. Die aufgespannte Fläche $A_x$ und die Länge $l_x$ eines Linienzuges $x$ werden dann durch die Funktion

$$f(x) = l_x - \alpha\sqrt{A_x}, \text{ hier mit } \alpha := 1 \tag{4.1}$$

bewertet. Über den Parameter $\alpha$ kann man die Gewichtung der Flächengröße bei der Bewertung kalibrieren. In den Feldtests lieferte der Wert $\alpha = 1$ gute Ergebnisse. Die Kandidaten der Referenzlinien werden anhand der Bewertung absteigend sortiert und der Benutzerschnittstelle bereitgestellt. In Tabelle 4.2 sind die Bewertungen der in Abbildung 4.6 dargestellten möglichen Referenzlinien mit Länge und Fläche aufgelistet.

**Tabelle 4.2:** Bewertung möglicher Referenzlinien aus Abbildung 4.6.

| **Nummer** | **Länge** $[m]$ | **Fläche** $[m^2]$ | **Bewertung** |
|---|---|---|---|
| 1 | 128,39 | 396,28 | 108,48 |
| 2 | 127,63 | 395,38 | 107,75 |
| 3 | 67,67 | 16,51 | 63,60 |
| 4 | 48,18 | 2,71 | 46,54 |

### 4.1.4 Berechnung der Fahrspuren

In diesem Abschnitt wird die geometrische Berechnung der Fahrspuren beschrieben, die von der Komponente *TrackServer* durchgeführt wird. Für die Berechnung werden eine Referenzlinie (siehe Abschnitt 4.1.3), die innere Feldumrandung, Umrandungen von statischen Hindernissen im Feld und die Arbeitsbreite der Erntemaschine benötigt. Die erstellten Fahrspuren müssen für die angegebene Arbeitsbreite die gesamte innere Feldfläche abdecken und sollen um im Voraus bekannte statische Hindernisse herum führen.

Das Vorgehen der Berechnung der Fahrspuren ist in Algorithmus 1 gegeben. Die Referenzlinie wird in jeder Iteration dreimal nach dem in Abbildung 4.7 gezeigten Vorgehen parallel verschoben. Dazu werden alle Segmente der Referenzlinie um die Verschiebungsdistanz orthogonal nach rechts verschoben. Aus den Segmenten und deren Verschiebungen werden Ringe konstruiert, die sukkzessiv zu einer Fläche vereinigt werden. Wenn sich zwei aufeinanderfolgende Ringe nicht schneiden, wird der Fläche vor der Schnittmengenbildung ein Kreis mit dem Radius der Verschiebungsdistanz um den Verbindungspunkt auf der Referenzlinie hinzugefügt. Aus der Umrandung der konstruierten Fläche wird dann die verschobene Referenzlinie zwischen dem verschobenen Start- und Endpunkt der Referenzlinie ausgeschnitten.

In jeder Iteration wird die Fläche zwischen den Linienzügen a und c als Ring $r_l$ erzeugt, der die abgedeckte Fläche der kompletten verschobenen Fahrspur repräsentiert (Abbildung 4.7). Diese geht über die Grenzen der innere Feldfläche hinaus und muss daher gekürzt werden. Zu diesem

**Algorithmus 1** Fahrspurberechnung aus Referenzlinie $l$, Arbeitsbreite $w$, innerer Feldumrandung $r_{inner}$ und statischen Hindernissen $O$

**Prozedur** FAHRSPURBERECHNUNG($l$, $w$, $r_{inner}$, $O$)
  initialisiere leeren Fahrspurvektor $T$
  initialisiere Laufindex $i = 0$
  **führe aus**
    verschiebe $l$ parallel um $i \cdot w$ und speichere als $a$
    verschiebe $l$ parallel um $(i + 0,5) \cdot w$ und speichere als $b$
    verschiebe $l$ parallel um $(i + 1) \cdot w$ und speichere als $c$
    erzeuge Ring $r_l$ aus Fläche zwischen $a$ und $c$ (Abb. 4.8)
    berechne Schnittflächen $X$ von $r_l$ und $r_{inner}$
    **für alle** Schnittflächen $x \in X$ **führe aus**
      berechne abgedeckte Fahrspurteile $S$ als Schnittmenge von $b$ und $x$
      **für alle** Fahrspurteile $s \in S$ **führe aus**
        verlängere Start und Ende von $s$, falls für Abdeckung von $x$ nötig (Abb. 4.8)
        ersetze alle ein Hindernis $o \in O$ schneidende Teile von $s$
        füge $s$ dem Fahrspurvektor $T$ hinzu
      **Ende für alle**
    **Ende für alle**
    inkrementiere Laufindex $i$ um 1
  **solange** mindestens eine neue Fahrspur gefunden wurde
  **gib zurück** Fahrspurvektor $T$
**Ende Prozedur**

Zweck werden die Schnittflächen von $r_l$ und der inneren Feldfläche gebildet. Die ermittelten Flächen beschreiben die tatsächlich abzuerntenden Flächen der gesamten verschobenen Fahrspur. Für jede dieser Flächen werden die abgedeckten Teile des Linienzugs $b$ bestimmt. Für alle diese Teile wird dann der Start und das Ende verlängert, falls die Fahrspur nicht orthogonal zur Feldgrenze endet. Dazu wird der Linienzug um die Projektion der Schnittpunkte auf $b$ für die betrachtete Fläche erweitert (siehe Abbildung 4.7).

Kollidiert die abgedeckte Fläche einer auf diese Weise extrahierten Fahrspur mit einem im Voraus bekannten Hindernis, so wird die Fahrspur um das Hindernis herumgeführt. Dazu wird die Kontur des Hindernisses um die halbe Arbeitsbreite aufgebläht und mit der Fahrspur verschnitten. Die Fahrspur wird an diesen Schnittpunkten aufgetrennt und in der kürzeren der zwei möglichen Strecken um das Hindernis geführt. Die so entstandenen Fahrspuren werden dem Fahrspurvektor hinzugefügt. Findet der Algorithmus in einer Iteration keine Fahrspur mehr, so terminiert der Algorithmus und liefert die gefundenen Fahrspuren zurück.

Zusätzlich zu den Fahrspuren, die die innere Feldfläche für die gegebene Arbeitsbreite des Mähdreschers vollständig abdecken, benötigt die Bewegungsplanung (Kapitel 5) eine Sollfahrstrecke in dem Vorgewende. Dabei wird ähnlich wie beim geometrischen Berechnen der inneren Feldfläche aus Abschnitt 4.1.1 vorgegangen. Die innere Feldgrenze wird um einen vorgegebenen Abstand aufgebläht. Dazu werden im Uhrzeigersinn alle Segmente der inneren Feldgrenze orthogonal nach

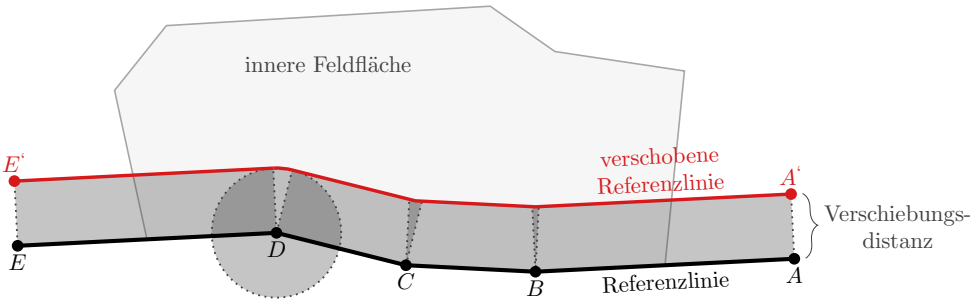



**Abbildung 4.7:** Parallelverschiebung einer Referenzlinie. Alle Segmente der Referenzlinie werden um die Verschiebungsdistanz orthogonal nach rechts verschoben. Aus den Segmenten und deren Verschiebungen werden Ringe konstruiert, die sukkzessiv zu einer Fläche vereinigt werden. Wenn sich zwei aufeinanderfolgende Ringe nicht schneiden, wird der Fläche erst ein Kreis mit dem Radius der Verschiebungsdistanz um den Verbindungspunkt auf der Referenzlinie hinzugefügt. In der Abbildung ist dies bei Punkt D der Fall. Aus der Umrandung der konstruierten Fläche wird dann die verschobene Referenzlinie zwischen dem verschobenen Start- und Endpunkt der Referenzlinie, hier also zwischen $A^{‘}$ und $E^{‘}$, ausgeschnitten.

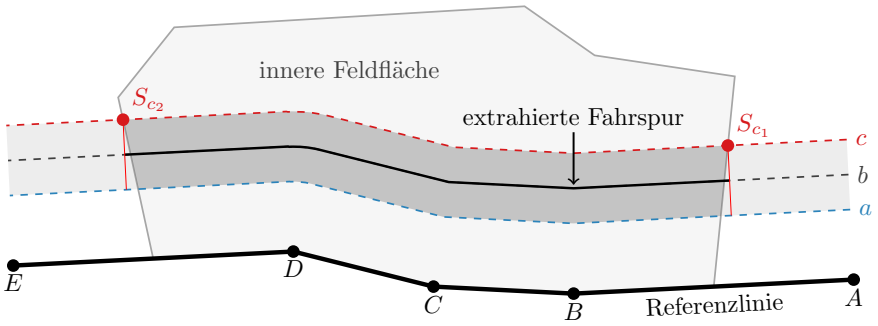



**Abbildung 4.8:** Schema der Fahrspurberechnung aus Parallelverschiebungen der Referenzlinie. Aus der Schnittmenge des Linienzuges $b$ und der Schnittfläche der inneren Feldfläche und der Fläche zwischen $a$ und $c$ wird die Fahrspur bestimmt. Der Linienzug $b$ wird um die Projektion der Schnittpunkte (hier $S_{c_1}$ und $S_{c_2}$) erweitert, um die gesamte Fläche abzudecken.

links verschoben. Die Rechtecke, die durch die Punkte der Segmente und der verschobenen Segmente beschrieben werden, werden schrittweise geometrisch vereint. Wenn aufeinanderfolgende Flächen sich dabei nicht überlappen, wird zusätzlich ein Kreis um den gemeinsamen Punkt der Segmente mit dem vorgegebenen Abstand als Radius vereint. Die äußere Kontur des so entstehenden Polygons ist die Vorgewendefahrspur. Im Folgenden wird der Begriff Fahrspur für die die innere Feldfläche abdeckenden Fahrspuren genutzt, während der Begriff Vorgewendefahrspur die Sollstrecke im Vorgewende beschreibt.

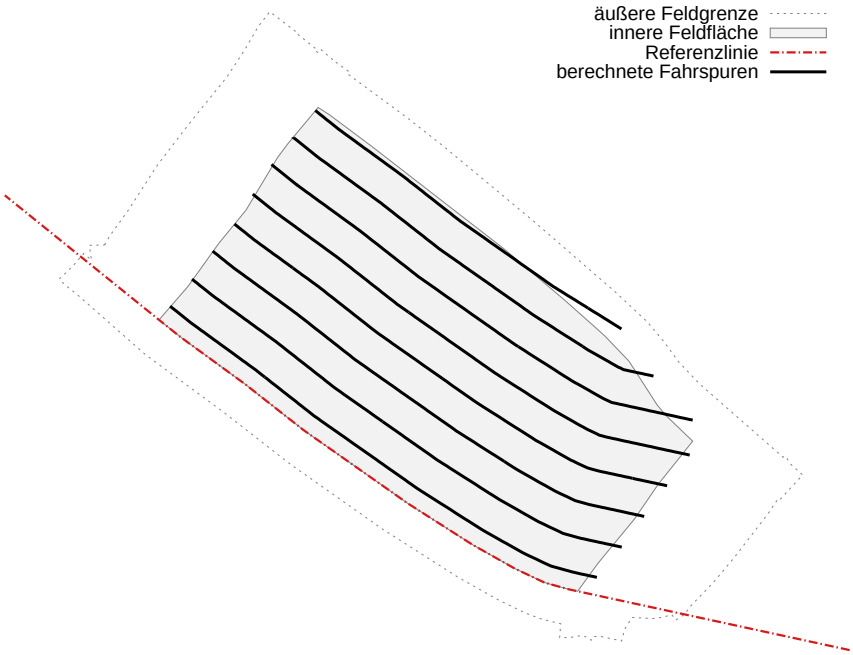



**Abbildung 4.9:** Ergebnis der Fahrspurberechnung für das aufgezeichnete Feld aus Abbildung 4.5(b) und einer der in Abbildung 4.6 ermittelten Referenzlinien. Die Fahrspuren decken die innere Feldfläche für die gegeben Arbeitsbreite des Mähdreschers vollständig ab.

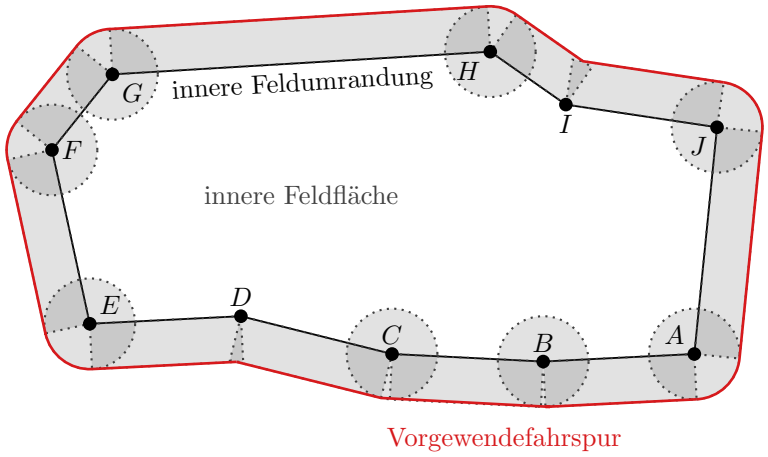



**Abbildung 4.10:** Geometrische Berechnung der Vorgewendefahrspur aus der inneren Feldumrandung und einem voreingestellten Abstand. Alle Segmente der inneren Feldumrandung werden im Uhrzeigersinn orthogonal nach außen verschoben. Die Flächen zwischen ursprünglichen und verschobenen Segmenten werden vereinigt. Dabei werden Kreise eingefügt, falls sich Flächen nicht überschneiden. Der äußere Ring der berechneten Fläche ergibt die Vorgewendefahrspur.

## 4.2 Ansatz der Routenplanung

Die Aufgabe der Routenplanung ist, wie in Abschnitt 2.2 beschrieben, die Planung des Ernte- und Transportprozesses für die Getreideernte. Für die Feldfahrzeuge müssen Routen berechnet werden, die diese Aufgaben erfüllen. Für den Mähdrescher besteht eine Route aus einer Fahrspurreihenfolge und aus Überladeprozessen. Die Fahrspurreihenfolge definiert den Ernteprozess, die Überladeprozesse definieren den Transportvorgang. Die Route für das Überladefahrzeug beschreibt die Überladeprozesse, bei denen der Mähdrescher seinen Korntankinhalt überlädt und Überladeprozesse am Feldrand, bei denen das Überladefahrzeug seine Ladung an einen Straßentransporter übergibt. Die Routen der Fahrzeuge beeinflussen sich gegenseitig in Raum und Zeit. Das Überladefahrzeug darf sich nur auf abgeernteter Fläche bewegen und ist daher auf die Wege angewiesen, die der Mähdrescher im Laufe des Erntevorgangs erzeugt. Der Mähdrescher wiederum ist darauf angewiesen, den Inhalt seines Korntanks rechtzeitig und während der Fahrt auf seiner Route in den Überladewagen überladen zu können.

Die Planungen des Ernte- und Transportprozesses beeinflussen sich demnach gegenseitig raumzeitlich und haben potentiell gegenläufige Optimierungsziele. Jede Planänderung von einem der beiden Prozesse beeinflusst den Plan des anderen Prozesses. Diese Abhängigkeiten führen zu einer hohen Komplexität der Optimierung des Gesamtprozesses in dem Infield-Transportlogistik Szenario. Bei einer erschöpfenden Suche nach dem globalen Optimum müssten für alle möglichen Teilpläne des Ernteprozesses alle möglichen Teilpläne des Transportprozesses betrachtet werden.

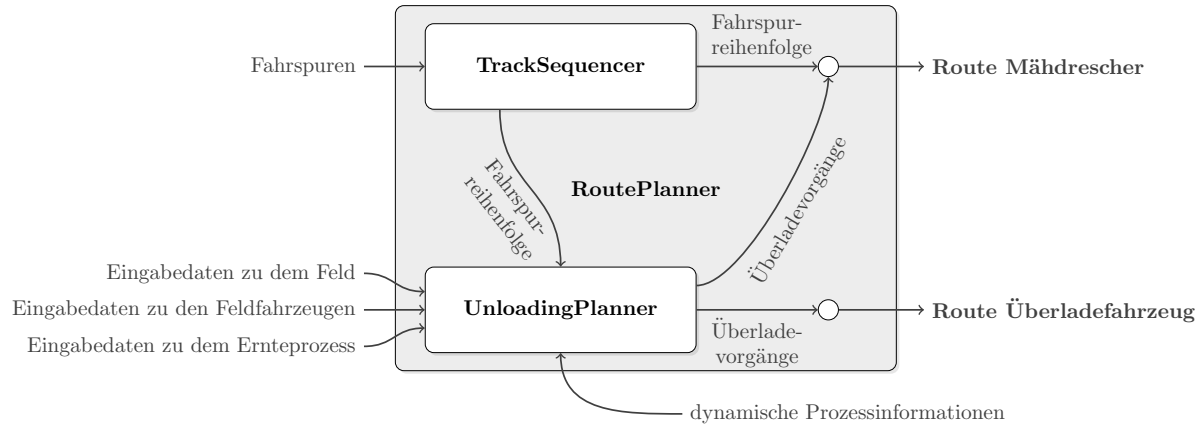



**Abbildung 4.11:** Ansatz der Routenplanung in der Planungssoftware, die in zwei Planungsschritte unterteilt wird. Die Komponente *TrackSequencer* legt die Fahrspurreihenfolge für den Mähdrescher fest. In einem nachgeordneten Planungsschritt werden von der Komponente *UnloadingPlanner* die Überladeprozesse des Mähdreschers und des Überladefahrzeugs geplant. Aus den Ergebnissen der zwei Planungsschritte werden die Routen für die Feldfahrzeuge erstellt.

Um die Komplexität der maschinenübergreifenden Planung zu verringern und um die gegenseitigen raumzeitlichen Abhängigkeiten von Ernte- und Transportprozess zu beantworten, wird die Routenplanung in zwei nacheinander zu lösende Planungsprobleme unterteilt. Der erste Planungsschritt ist die Bestimmung der Fahrspurreihenfolge für den Mähdrescher. Der zwei-

te, nachgeordnete Planungsschritt ist die Bestimmung der Überladeprozesse für Mähdrescher und Überladefahrzeug. Die Überladeplanung berücksichtigt dynamische Prozessinformationen, um bei Bedarf die Überladevorgänge und somit die Routen für die Feldfahrzeuge neu zu planen. In Abschnitt 4.3 wird die Bestimmung der Fahrspurreihenfolge für den Mähdrescher, in Abschnitt 4.4 die Bestimmung der Überladeprozesse für die beiden Feldfahrzeuge detailliert beschrieben.

Der vorgestellte Ansatz der Routenplanung unterscheidet sich von den in der Literatur beschriebenen Verfahren (siehe Abschnitt 3.1.6). In [15] wurde eine hierarchische Architektur zur Planung für Feldfahrzeuge vorgeschlagen, bei der die Planung von Fahrrouten als Instanz des Vehicle Routing Problems (VRP) formuliert wird. In [13, 14] wird der Vorschlag konkretisiert, die Aufgabenstellung der Infield-Transportlogistik als VRP zu definieren. Eine wesentliche Einschränkung ist jedoch, dass für diese Formulierung die Überladeorte a priori bekannt sein müssen. Weil die Wahl geeigneter Überladeprozesse für das beschriebene Szenario jedoch integraler Bestandteil der Planung sein muss, werden diese in dem vorgestellten Ansatz unter Berücksichtigung von prozessspezifischen Einschränkungen raumzeitlich geplant und optimiert.

## 4.3 Planung von Fahrspurreihenfolgen

Die Wahl der Fahrspurreihenfolge geschieht in der Praxis durch den Fahrer des Mähdreschers. Dabei wird meistens nach Strategien gefahren, bei denen möglichst oft die zur Fahrtrichtung linke benachbarte Fahrspur bereits abgeerntet wurde. Das ist wichtig, weil das Korntankrohr nach links ausschwenkt und sich das Überladefahrzeug nur auf bereits abgeernteter Fläche bewegen darf. Ist die linke benachbarte Fahrspur bereits abgeerntet, so kann der Korntankinhalt in Parallelfahrt in den Überladewagen überladen werden. Diese Strategien sind immer auf eine bestimmte Anzahl an Fahrspuren ausgelegt. Ein Satz an Fahrspuren wird als Beet, die Anzahl der Fahrspuren in einem Beet als Beetbreite bezeichnet.

Um das Planungsproblem der Fahrspurreihenfolge algorithmisch lösen zu können, werden die Fahrspuren in einem topologischen Graphen modelliert. In Abbildung 4.12(a) sind die Fahrspuren aus dem Beispiel aus Abbildung 4.9 in einem solchen Graphen als Kanten dargestellt. Die Endpunkte der Fahrspuren sind entsprechend als Knoten repräsentiert. Die Knoten derselben Fahrspurenden werden, wie in Abbildung 4.12(a) gezeigt, untereinander vollständig verbunden. In dem Graphen $G = (V, E)$ gibt es

$$|V| = 2n$$

Knoten und

$$|E| = 2 \sum_{i=1}^{n-1} i + n$$

Kanten, von denen $n$ Kanten die $n$ abzuarbeitenden Fahrspuren repräsentieren.

Die Gewichte der Kanten, die Fahrspuren darstellen, beschreiben die Längen der Fahrspuren. Die anderen Kanten beschreiben die euklidischen Distanzen zwischen den Endpunkten der Fahrspuren. Die Knoten werden fahrspurweise so durchnummeriert, dass jeweils die durch gerade

und ungerade Zahlen beschriebenen Knoten an demselben Fahrspurende liegen (siehe Abbildung 4.12(a)). Die Kante zwischen Knoten 0 und 1 beschreibt also die Fahrspur 0, die Kante zwischen Knoten 2 und 3 die Fahrspur 1 und so weiter. In diesem Graphen kann eine Fahrspurreihenfolge gesucht und durch eine Knotensequenz kodiert werden. Eine Fahrspurreihenfolge ist gültig, wenn alle Fahrspuren genau einmal besucht werden. Wenn an den Fahrspurenden nur direkte Verbindungen zwischen den Knoten von aufeinander folgenden Fahrspuren zugelassen werden, besteht eine Fahrspurreihenfolge aus einer Folge von Knotenpaaren, dessen Ganzzahlendivision mit dem Divisor 2 gleich ist. Diese Knotenpaare kodieren Fahrspur und Fahrtrichtung. In (4.2) - (4.4) sind die Knotensequenzen für die Abbildungen 4.12(b) - 4.12(d) gegeben.

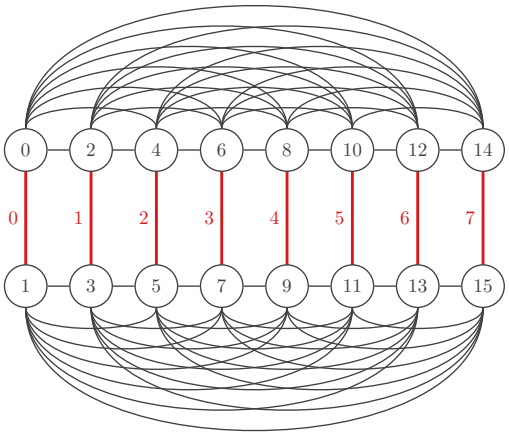
$$\underset{\text{Fahrspurnummer}}{Knotensequenz_1} = [\underbrace{0,1}_{0}, \underbrace{3,2}_{1}, \underbrace{4,5}_{2}, \underbrace{7,6}_{3}, \underbrace{8,9}_{4}, \underbrace{11,10}_{5}, \underbrace{12,13}_{6}, \underbrace{15,14}_{7}] \tag{4.2}$$

$$\underset{\text{Fahrspurnummer}}{Knotensequenz_2} = [\underbrace{6,7}_{3}, \underbrace{9,8}_{4}, \underbrace{4,5}_{2}, \underbrace{11,10}_{5}, \underbrace{2,3}_{1}, \underbrace{13,12}_{6}, \underbrace{0,1}_{0}, \underbrace{15,14}_{7}] \tag{4.3}$$

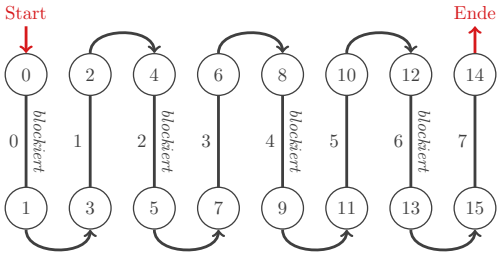
$$\underset{\text{Fahrspurnummer}}{Knotensequenz_3} = [\underbrace{8,9}_{4}, \underbrace{1,0}_{0}, \underbrace{6,7}_{3}, \underbrace{11,10}_{5}, \underbrace{4,5}_{2}, \underbrace{13,12}_{6}, \underbrace{2,3}_{1}, \underbrace{15,14}_{7}] \tag{4.4}$$

Eine Fahrspurreihenfolge kann angesehen werden als Permutation der Zahlen $(0, 1, \ldots, n-1)$ und der Information zu der Fahrtrichtung. Es existieren also $2 \cdot n!$ mögliche Fahrspurreihenfolgen für $n$ Fahrspuren.

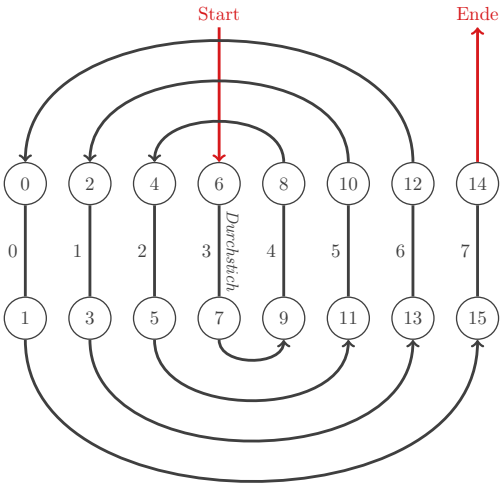
In Abbildung 4.12(b) ist die Fahrspurreihenfolge für den Mähdrescher mit der kürzesten Gesamtdistanz dargestellt. Die Fahrspuren werden der Reihe nach in gegenläufiger Richtung abgeerntet. Nachteil bei dieser Reihenfolge ist, dass auf jeder zweiten Fahrspur kein Überladeprozess stattfinden kann, weil die zur Fahrtrichtung linke Fahrspur noch nicht abgeerntet wurde. Wie einleitend beschrieben, wird in der Praxis deshalb nach bestimmten Mustern gefahren. Abbildung 4.12(c) zeigt exemplarisch ein Abarbeitungsmuster, bei dem nur auf einer Fahrspur, nämlich bei dem Durchstich auf Fahrspur 3, nicht abgetankt werden kann. Ansonsten wird kreisförmig so gefahren, dass immer die linke benachbarte Fahrspur bereits abgeerntet wurde. Der Nachteil dieser Strategie ist, dass durch die Fahrten im Vorgewende die Gesamtdistanz der zu fahrenden Strecke und damit auch die Prozesszeit zunimmt. Daher unterteilt man das Feld in Beete, in denen jeweils ein Durchstich durchgeführt werden muss, um die zu fahrende Gesamtstrecke praktikabel zu halten. Aufgrund der Performanz des für den Mähdrescher eingesetzten Lenksystems muss bei einem Fahrspurwechsel mindestens eine Fahrspur freigelassen werden. Deshalb wird für die Planung die Strategie aus Abbildung 4.12(d) mit einer Beetbreite von 8 Fahrspuren genutzt. Bei dieser Strategie wird beim Wenden immer mindestens eine Fahrspur übersprungen. Die Fahrspurreihenfolge ist in dem Planungssystem als festes Muster in der Komponente *TrackSequencer* hinterlegt und wird über die gesamte Anzahl an Fahrspuren geschoben. Wenn ein Beet keine 8 Fahrspuren hat, werden vordefinierte Sequenzen für die Restbeete gewählt. Ausgabe der Komponente ist ein Ganzzahlenvektor, der die Knotensequenz wie in (4.2) - (4.4) beschreibt.



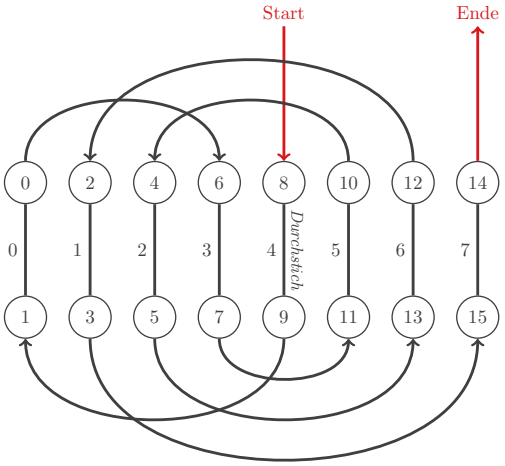

(a) Repräsentation der Fahrspuren durch Kanten (rot) in einem topologischen Graphen.




(b) Fahrspurreihenfolge mit kürzester Gesamtdistanz. Der Fahrweg für das Überladefahrzeug ist bei der Hälfte der Fahrspuren durch den Bestand blockiert.




(c) Beetstrategie mit nur einem Durchstich für acht Fahrspuren.




(d) Im Planungssystem genutzte Beetstrategie mit nur einem Durchstich für acht Fahrspuren.

**Abbildung 4.12:** Topologischer Graph und Strategien für Planung der Fahrspurreihenfolge. Die abzuarbeitenden Kanten sind in dem topologischen Graphen in (a) rot eingefärbt. Die Fahrspurreihenfolge mit der geringsten Gesamtdistanz ist in (b) gegeben. In dieser Reihenfolge kann allerdings nur auf jeder zweiten Fahrspur abgetankt werden. Bei den Beetstrategien aus (c) und (d) kann nur auf einer Fahrspur, bei dem sogenannten Durchstich nicht abgetankt werden. Allerdings ist dort die gefahrene Gesamtdistanz größer. Die in dem entwickelten Planungssystem eingesetzte Strategie aus (d) lässt beim Wenden immer mindestens eine Fahrspur aus, damit das Lenksystem die Wende durchführen kann.

## 4.4 Planung von Überladeprozessen

Für die Planung der Überladeprozesse werden die in Abschnitt 4.1 beschriebenen Eingaben und die in Abschnitt 4.3 beschriebene Fahrspurreihenfolge des Mähdreschers benötigt. Anhand dieser Daten können die relevanten Informationen für die Überladeplanung berechnet werden. Aus den durchschnittlichen Geschwindigkeiten und der Fahrspurreihenfolge für den Mähdrescher können dessen Position und die Gesamtmenge an geerntetem Getreide und die verbleibende Bestandsfläche für jeden Prozesszeitpunkt berechnet werden. Der Füllstand des Mähdreschers berechnet sich bei der Planung von Überladeprozessen aus den vor diesem Zeitpunkt liegenden Überladeprozessen selbst. Die Überladeplanung stellt dementsprechend ein nichtlineares Optimierungsproblem dar. Das Planungsproblem wird in Abschnitt 4.4.3 detaillierter vorgestellt und formalisiert. Um dieses Problem effizient in annehmbarer Zeit während des Ernteprozesses auf einem Fahrzeugrechner lösen zu können, wird es in ein diskretes Optimierungsproblem überführt. Dazu wird auch in diesem zweiten Schritt der Routenplanung ein topologischer Graph aufgebaut. Die Erzeugung des Graphen wird in Abschnitt 4.4.1 beschrieben. Anschließend wird in Abschnitt 4.4.2 die Einbindung von Ertragsprognosekarten für eine genauere Planung erläutert. Die Aufgabenstellung der Überladeplanung wird in Abschnitt 4.4.3 formal dargestellt, bevor in Abschnitt 4.4.4 eine abstrakte, von den Spezifika des Infield-Transportlogistik Szenarios losgelöste Problemstellung formuliert wird. Abschnitt 4.4.5 erklärt den implementierten und in Ernteversuchen getesteten Ansatz der Überladeplanung per Baumsuche.

### 4.4.1 Erzeugung eines topologischen Graphen für die Planung

Der Graph für die Überladeplanung wird aus der Feldgeometrie erzeugt. Dazu werden die Fahrspuren und der Abtankort am Feldrand benötigt. Anhand der geometrischen Eigenschaften dieser Eingabedaten werden Punkte identifiziert, die in dem Graphen durch Knoten repräsentiert werden. Diese Knoten sind schematisch für ein Beispiel in Abbildung 4.13 als schwarze Kreise eingezeichnet.

Diese Punkte beinhalten eine diskrete Darstellung der abzuarbeitenden Fahrspuren und der Vorgewendefahrspur und außerdem den Abtankort am Feldrand. Zur Bestimmung der Punkte für die abzuarbeitenden Fahrspuren, in dem Beispiel also Fahrspur eins bis vier, werden die Linienzüge nacheinander betrachtet. Auf der ersten Fahrspur wird alle fünf Meter ein Punkt eingefügt und in einer Liste gespeichert. Zusätzlich wird der Endpunkt der Fahrspur der Liste hinzugefügt. Für alle diese Punkte wird der Schnittpunkt der Orthogonalen in diesem Punkt der Fahrspur mit der nächsten Fahrspur berechnet. Diese Punkte werden für die zweite Fahrspur in einer Liste vorgemerkt. Start- und Endpunkte werden zusätzlich in die Liste aufgenommen, falls diese noch nicht in dieser vorhanden sind. Die Distanz auf der zweiten Fahrspur zwischen den Punkten wird dann auf einen minimalen und auf einen maximalen Wert hin überprüft. Falls zwei Punkte zu nahe beieinander liegen, werden diese durch einen mittig zwischen den Punkten liegenden Punkt ersetzt. Falls die Distanz zu groß ist, so wird mittig zwischen den Punkten ein zusätzlicher Punkt eingefügt. Beide Fälle treten in dem Beispiel in Abbildung 4.13 auf der dritten Fahrspur bei den Punkten $P_{2,5}$ und $P_{2,7}$ auf. In diesem Beispiel wird für alle weiteren

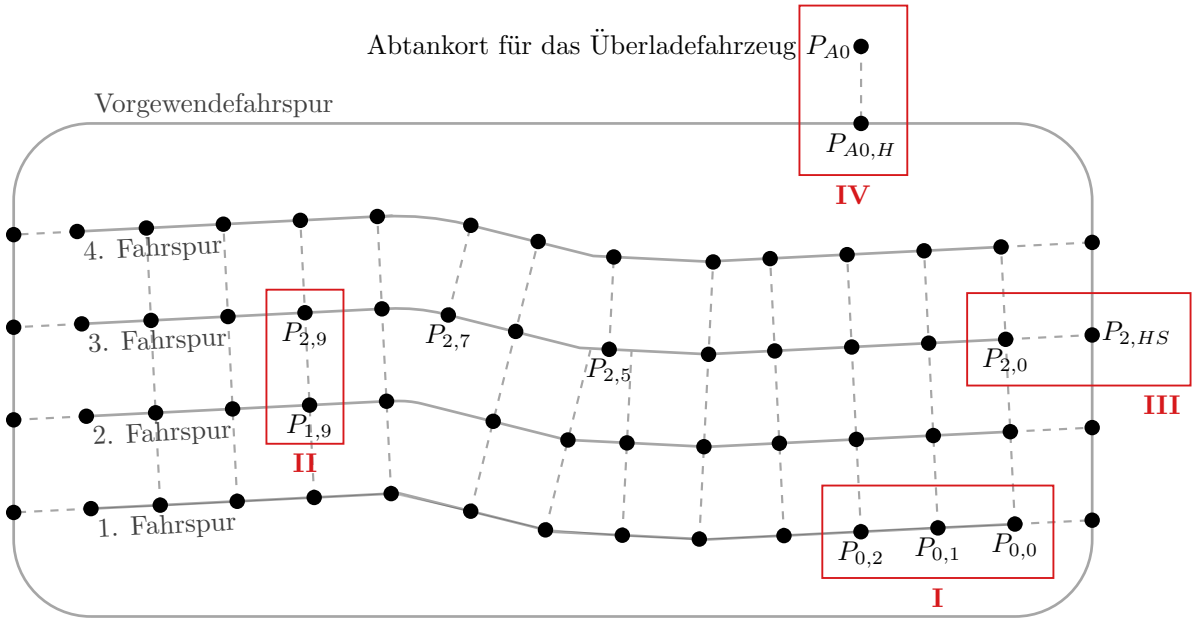



**Abbildung 4.13:** Grundlage der Erstellung des Graphen für die Überladeplanung sind die Fahrspuren, die Vorgewendefahrspur und die Abtankorte für das Überladefahrzeug am Feldrand. Anhand dieser Geometrien werden relevante Punkte (schwarz) identifiziert, die in dem Graphen durch Knoten repräsentiert werden (siehe Abbildung 4.14).

Fahrspuren wie für Fahrspur zwei vorgegangen. Nachdem auf diese Weise für alle Fahrspuren die relevanten Punkte ermittelt wurden, werden die Verbindungspunkte zwischen den abzuarbeitenden Fahrspuren und der Fahrspur im Vorgewende bestimmt. Dazu wird für jede Fahrspur der Schnittpunkt der Vorgewendefahrspur mit den nach außen verlängerten ersten und letzten Fahrspursegmenten berechnet. Anschließend wird der Abtankort am Feldrand und der zu dem Abtankort nächste Punkt auf der Vorgewendefahrspur vorgemerkt. An dieser Stelle sind alle relevanten Punkte erfasst. Diese müssen nun durch Knoten in einem gerichteten Graphen repräsentiert und durch Kanten so miteinander verbunden werden, dass in dem Graphen die Überladeplanung durchgeführt werden kann.

Dabei werden alle Punkte der Fahrspuren durch zwei Knoten repräsentiert, um die unterschiedlichen Fahrtrichtungen darzustellen. In Abbildung 4.14(a) ist exemplarisch die Verbindung der ersten drei Punkte ($P_{0,0}, P_{0,1}, P_{0,2}$) der ersten Fahrspur aus dem Ausschnitt I der Abbildung 4.13 gezeigt. Das Knotenpaar, das einen Punkt einer Fahrspur repräsentiert, hat immer aufeinanderfolgende Identifikationsnummern. Die geraden und ungeraden Knoten stellen die unterschiedlichen Fahrtrichungen dar. Die Fahrspurknoten werden daher so verbunden, dass alle geraden und alle ungeraden Knoten miteinander in derselben Richtung verbunden werden. Die geraden Knoten zeigen in dem konstruierten Graphen in dieselbe Richtung, wie der zugrunde liegende Linienzug der Fahrspur. Daher sind die Kanten zwischen den Knoten 0, 2 und 4 in Abbildung 4.14(a) als *vorwärts* und die Kanten zwischen den Knoten 5, 3 und 1 als *rückwärts* beschriftet.

Für die Planung von Überladeprozessen müssen den Fahrspurknoten benachbarte Fahrspurknoten als solche zugeordnet werden können. Das ist nötig, weil sich das Überladefahrzeug bei einem Überladeprozess auf der linken benachbarten Fahrspur in gleicher Fahrtrichtung wie der Mähdrescher bewegen muss. Diese Verbindungen zwischen benachbarten Fahrspurknoten ist schematisch in Abbildung 4.14(b) für den Ausschnitt II mit den Punkten $P_{1,9}$ und $P_{2,9}$ in Abbildung 4.13 gezeigt. Die Knoten der benachbarten Fahrspuren werden vollständig untereinander verbunden, um den Wechsel von Fahrspuren in dieselbe und in die entgegengesetzte Fahrtrichtung für die Fahrwegsplanung des Überladefahrzeugs zu ermöglichen.

Die Verbindung der Fahrspuren mit der Vorgewendefahrspur wird in Abbildung 4.14(c) als Ausschnitt III der Abbildung 4.13 mit den Punkten $P_{2,0}$ und $P_{2,HS}$ gezeigt. Auf der Vorgewendefahrspur gibt es ebenfalls zwei Fahrtrichtungen, daher ist die Verbindung der Fahrspur mit der Vorgewendefahrspur in dem Graphen mit einer T-Kreuzung in einem Straßennetz vergleichbar. Es werden sechs Knoten für den Schnittpunkt $P_{2,HS}$ erzeugt, die wie in Abbildung 4.14(c) verbunden werden. An diesen Verbindungen sind Wendemanöver für alle Fahrtrichtungen gültig. Die Knoten der Vorgewendefahrspur werden jeweils in derselben Fahrtrichtung und somit analog zu den abzuarbeitenden Fahrspuren verbunden.

Die Abtankorte für das Überladefahrzeug werden durch einen Knoten repräsentiert und mit zwei eingefügten Knoten der Vorgewendefahrspur verbunden, wie exemplarisch in Abbildung 4.14(d) als Ausschnitt IV der Abbildung 4.13 dargestellt. Aus jeder Fahrtrichtung im Vorgewende ist der Knoten des Überladepunkts erreichbar. Von diesem Knoten aus kann das Überladefahrzeug auch in beide Fahrtrichtungen auf die Vorgewendefahrspur gelangen.

Die Knoten des Graphen haben die Attribute

- **Identifikationsnummer** $\in \mathbb{N}$, mit der fortlaufend alle Knoten durchnummeriert werden.
- **Fahrspurnummer**, zu der ein Knoten gehört. Für Knoten von den $n$ abzuerntenden Fahrspuren ist dieser Wert $\in \{0, \ldots, n-1\}$, für Knoten der Vorgewendefahrspur ist dieser Wert $-1$, für den Überladepunkt ist der Wert $-2$.
- **2D Koordinaten** $\in \mathbb{R}^2$ des durch den Knoten repräsentierten Punkts.

Diese Attribute werden bei dem Erzeugen des topologischen Graphen berechnet. Die Kanten des Graphen haben die Attribute

- **Quellknoten** der Kante.
- **Zielknoten** der Kante.
- **Distanz** $\geq 0$ als Gewicht der Kante.
- **Ertrag** der durch die Kante repräsentierten Fläche. $\geq 0$ für Fahrspurkanten, sonst 0.
- **Zeitfenster**, zu denen die Kante durch Bestand oder ein anderes Fahrzeug blockiert ist.

Die Distanzen der Kanten entprechen den Distanzen der jeweils repräsentierten Fahrspurabschnitte. Den Kanten, die Wendemanöver kodieren, wird ein vordefinierter Wert als Distanz

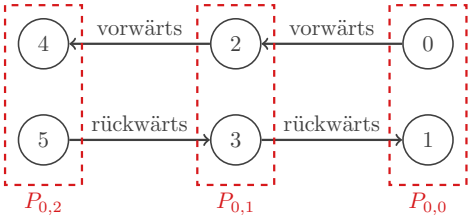
zugewiesen. Für alle übrigen Kanten wird der euklidische Abstand der verbundenen Knoten als Distanz berechnet.

Der Ertrag der Fahrspurkanten kann als Produkt der Distanz, der Arbeitsbreite des Mähdreschers und des geschätzten Ertrages für das Feld berechnet werden. Alternativ können die Erträge der durch eine Kante repräsentierten Fläche mit dem in Abschnitt 4.4.2 beschriebenen Vorgehen präziser geschätzt werden.

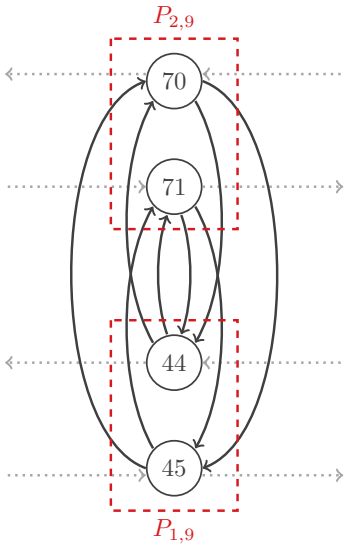
Für die Berechnung der Zeitfenster werden die Fahrspurreihenfolge des Mähdreschers und seine Durchschnittsgeschwindigkeiten für das Ernten und Wenden benötigt. Die Zeitfenster müssen bei Abweichungen in der Planausführung neu berechnet werden. Die Zeitfenster kodieren die nicht abgeerntete Feldfläche und den Mähdrescher als dynamische Hindernisse für das Überladefahrzeug. Mit Hilfe der Zeitfenster können die kürzesten Fahrwege des Überladefahrzeugs in der Überladeplanung zu einem beliebigen Prozesszeitpunkt berechnet werden. So kann unter anderem festgestellt werden, ob das Überladen an einem bestimmten Knoten zu einer bestimmten Zeit überhaupt möglich ist und ob das Überladefahrzeug den entsprechenden Knoten rechtzeitig erreichen kann.

Aus der Fahrspurreihenfolge des Mähdreschers kann die Knotenreihenfolge in dem beschriebenen Graphen ermittelt werden. Für jeden dieser Knoten kann die insgesamt abgeerntete Erntemasse und die verstrichene Prozesszeit bei dem Erreichen des Knotens durch den Mähdrescher berechnet werden.

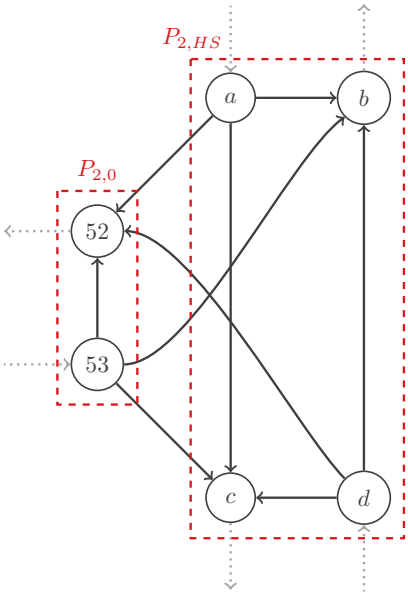
Die Nutzung topologischer Darstellungen der Umgebung ist in der mobilen Robotik durchaus üblich (siehe Abschnitt 3.1.3), weil sie effizientes Planen ermöglicht [42]. In der Literatur zur landwirtschaftlichen Robotik werden häufig topologische Karten zur Repräsentation eines Felds genutzt, beispielsweise in [4, 15, 45] (siehe Abschnitt 3.1.6). Neu an der vorgestellten Art der Repräsentation eines Felds in einem topologischen Graphen ist die Ausprägung und die Art der Generierung des Graphen aus der Feldgeometrie. Der Graph enthält eine gerichtete, diskretisierte Darstellung der Fahrspuren im inneren Feldbereich und im Vorgewende. Durch die Verbindungen der Fahrspuren untereinander kann für ein Überladefahrzeug unter Berücksichtigung von Zeitfenstern und dem Startzeitpunkt ein kürzester Pfad bestimmt werden, der das noch nicht abgeerntete Feld als dynamisches Hindernis berücksichtigt. In dem erstellten Graphen kann eine Überladeplanung für den in Abschnitt 2.2 beschrieben Anwendungsfall der Infield-Transportlogistik durchgeführt werden.



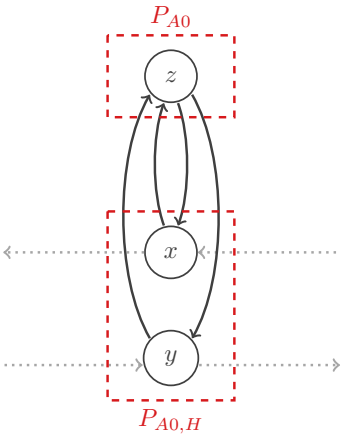

(a) Verbindung von Punkten derselben Fahrspur. Ausschnitt I aus Abbildung 4.13.




(b) Verbindung von Punkten benachbarter Fahrspuren. Ausschnitt II aus Abbildung 4.13.




(c) Verbindung von Fahrspur im Feld zur Fahrspur im Vorgewende. Ausschnitt III aus Abbildung 4.13.




(d) Verbindung von Abtankort für das Überladefahrzeug mit der Fahrspur im Vorgewende. Ausschnitt IV aus Abbildung 4.13.

**Abbildung 4.14:** Schematische Darstellung der Konstruktion des topologischen Graphen zur Überladeplanung für das Beispiel aus Abbildung 4.13.

### 4.4.2 Einbindung von Ertragsprognosekarten

Für die Planung von Ernteprozessen liefern verschiedene Anbieter satellitengestützte Ertragsprognosekarten. Je nach Genauigkeit der Karten könnten Ernteprozesse genauer, also mit weniger Abweichungen in der Prozessausführung geplant werden. Daher wurde die Einbindung solcher Karten in das Planungssystem ermöglicht. Die Eingabedaten der Ertragsprognosekarten müssen folgende Informationen enthalten.

- **Anzahl der Spalten** in der Rasterkarte
- **Anzahl der Zeilen** der Rasterkarte
- **Ursprung** der Rasterkarte in UTM Koordinaten
- **Datentabelle** mit Erträgen für alle Zellen

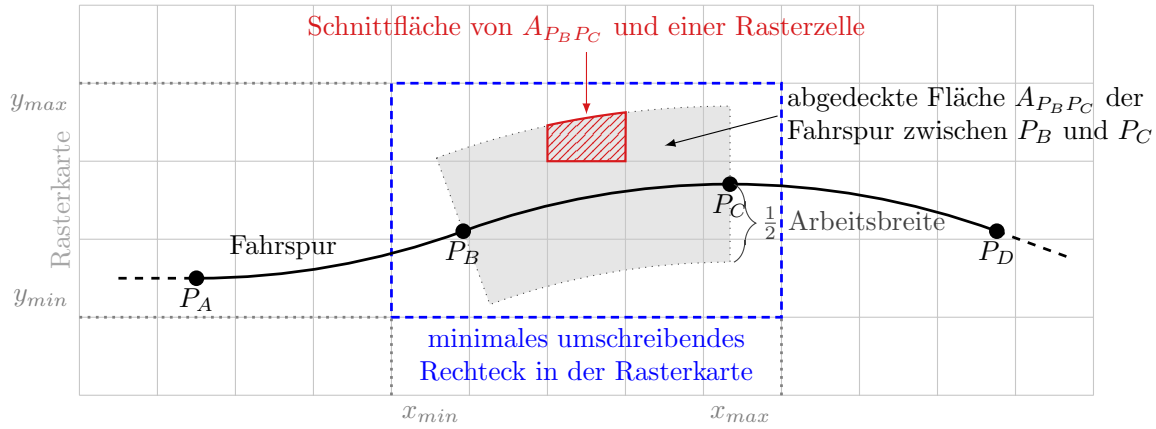



**Abbildung 4.15:** Berechnung von Erträgen für Fahrspurabschnitte. Die mit der Arbeitsbreite des Mähdreschers abgedeckte Fläche $A_{P_B P_C}$ zwischen den Punkten $P_B$ und $P_C$ wird mit allen Gitterzellen in dem minimal umschreibenden Rechteck um $A_{P_B P_C}$ geometrisch verschnitten. Alle Schnittflächen werden mit den entsprechenden Zellwerten der Ertragsprognosekarte multipliziert und aufsummiert. Das Ergebnis ist der Ertrag laut Ertragsprognosekarte für das Fahrspurstück.

Die Erträge werden geometrisch mit Hilfe dieser Karte berechnet und als Attribute der Fahrspurkanten in dem in Abschnitt 4.4.1 beschriebenen topologischen Graphen gespeichert.

Für jede Fahrspur wird mit der Arbeitsbreite des Mähdreschers die von dem Fahrspurabschnitt zwischen zwei Punkten abgedeckte Fläche berechnet. In Abbildung 4.15 ist das exemplarisch für die Fläche $A_{P_B P_C}$ des Fahrspurabschnitts zwischen den Punkten $P_B$ und $P_C$ dargestellt. Für diese Fläche wird das minimale umschreibende Rechteck in der Rasterkarte ermittelt. Alle in diesem Rechteck liegenden Zellen werden mit der Fläche $A_{P_B P_C}$ des Fahrspurabschnitts verschnitten und mit dem Ertragswert der Zelle multipliziert. Die Summe dieser Werte ergibt die tatsächlich abzuerntende Masse für den Fahrspurabschnitt. Der Wert wird in dem Attribut *Ertrag* der beiden Kanten gespeichert, die diesen Fahrspurabschnitt repräsentieren. Der Ertrag

$y$ der Fahrspurfläche $A_x$ berechnet sich demnach als

$$y(A_x) = \sum_{i=x_{min}}^{x_{max}} \sum_{j=y_{min}}^{y_{max}} (A_{ij} \cup A_x) \cdot e_{i,j} \tag{4.5}$$

mit dem Ertragswert $e_{ij}$ und der Fläche $A_{ij}$ einer Zelle $(i, j)$ der Rasterkarte.

### 4.4.3 Formale Beschreibung der Planung von Überladeprozessen

Für die Planung von Überladeprozessen wird die Knotenreihenfolge des Mähdreschers in dem topologischen Graphen aus Abschnitt 4.4.1 identifiziert. Alle $k$ besuchten Fahrspurknoten werden in der Abarbeitungsreihenfolge des Mähdreschers von 0 bis $k-1$ durchnummeriert (siehe Abbildung 4.16). Die Knotenreihenfolge ist durch die gewählte Fahrspurreihenfolge des Mähdreschers aus Abschnitt 4.3 gegeben. Zu jedem besuchten Knoten werden zusätzliche Attribute gespeichert, die für die Überladeplanung relevant sind. In Tabelle 4.4 sind diese Attribute aufgelistet. Jeder dieser Knoten ist ein potentieller Startpunkt für einen Überladeprozess. Das Ergebnis der Überladeplanung ist eine Folge von Knoten $x_0, x_1, \ldots, x_{|x|-1}$, an denen Überladeprozesse starten. Die überladene Menge und die Dauer des Überladeprozesses ergeben sich rechnerisch aus den Attributen eines Knotens und vorhergehenden Überladeprozessen.

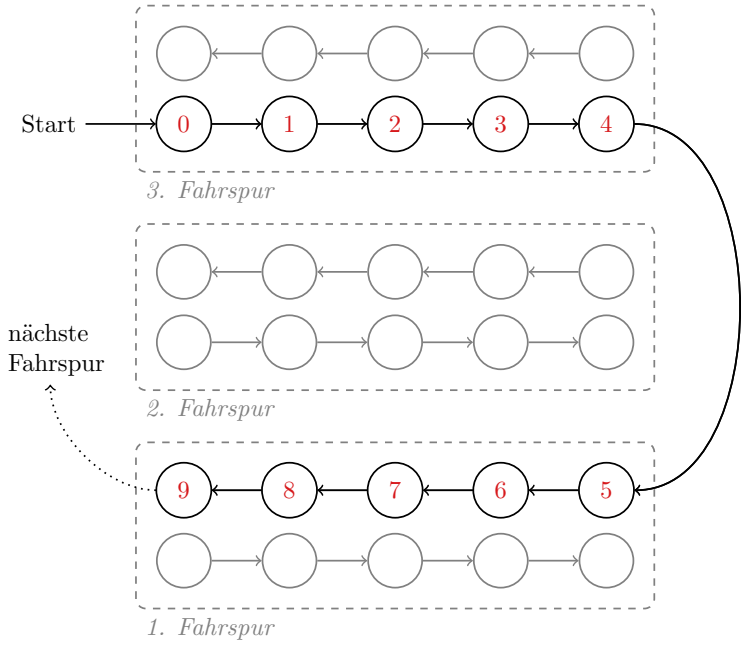



**Abbildung 4.16:** Nummerierung der Knoten anhand der Fahrspurreihenfolge. Die von dem Mähdrescher befahrenen Knoten des topologischen Graphen werden identifiziert und der Reihe nach aufsteigend nummeriert. In diesem Beispiel ist das schematisch für zwei Fahrspuren einer willkürlich gewählten Fahrspurreihenfolge gezeigt. Für die nummerierten Knoten werden die Knotenattribute aus Tabelle 4.4 berechnet, die für die Planung von Überladevorgängen benötigt werden.

**Tabelle 4.3:** Maschinen-, feld- und prozessspezifische Parameter für die Überladeplanung. Die Kapazitäten und Abtankleistungen der Fahrzeuge, die Anzahl der Fahrspuren des Feldes und die prozessspezifischen Schwellwerte bilden die Grundlage der Planung.

| Symbol | Beschreibung |
|---|---|
| $\alpha$ | Faktor $> 0$ für Berechnung der zu überladenen Menge |
| $\beta$ | maximaler Füllstand $\in [0, 100]$ für den Korntank des Mähdreschers in Prozent |
| $\gamma$ | prozentualer Füllstand $\in [0, 100]$ für den Korntank des Mähdreschers, bis zu dem immer überladen wird |
| $\delta$ | prozentualer Füllstand $\in [0, \beta]$ für den Korntank des Mähdreschers, bei dem Überladeprozesse starten sollen |
| $n$ | Anzahl $\in \mathbb{N}$ an Fahrspuren |
| $c_h$ | Kapazität $\geq 0$ des Korntanks für den Mähdrescher |
| $c_{ulv}$ | Kapazität $\geq 0$ des Korntanks für das Überladefahrzeug |
| $p_h$ | Abtankleistung $> 0$ des Mähdreschers |
| $p_{ulv}$ | Abtankleistung $> 0$ des Überladefahrzeugs |

**Tabelle 4.4:** Attribute der Knoten, die vom Mähdrescher befahren werden. Die Attribute sind abhängig von der für den Mähdrescher gewählten Fahrspurreihenfolge.

| Symbol | Beschreibung |
|---|---|
| $v_i$ | Identifikator $\in \mathbb{N}$ des Knotens $i$ in dem topologischen Graphen |
| $t_i$ | Zeitpunkt $\geq 0$, zu dem Knoten $i$ vom Mähdrescher erreicht wird |
| $m_i$ | Gesamtmasse $\geq 0$, die bis Knoten $i$ geerntet wurde |
| $track_i$ | Fahrspurnummer $\in \{0, \ldots, n-1\}$ von Knoten $i$ |
| $valid_i$ | Merker $\in \{0, 1\}$, ob der Knoten $i$ geometrisch als Start für einen Überladeprozess zulässig ist |

Da der Mähdrescher während des Überladens in der Parallelfahrt weiter erntet, muss die dabei aufgenommene Erntemasse bei der Berechnung der zu überladenen Masse berücksichtigt werden. Diese berechnet sich anhand des Korntankfüllstands $l$, der Arbeitsbreite $w$ und der durchschnittlichen Erntegeschwindigkeit $v_h$ des Mähdreschers und dem durchschnittlichen Ertrag $y$ des Feldes als

$$\underbrace{\frac{l}{p_h}}_{\text{Überladedauer}} \cdot \underbrace{y \cdot v_h \cdot w}_{\text{pro Sekunde aufgenommene Masse}} . \tag{4.6}$$

Diese Masse muss also zusätzlich überladen werden. In der dazu benötigten Zeit wird ebenfalls weiter geerntet. Die dabei aufgenommene Masse berechnet sich als

$$\underbrace{\frac{l \cdot y \cdot v_h \cdot w}{p_h}}_{\text{Gleichung 4.6}} \cdot \underbrace{\frac{y \cdot v_h \cdot w}{p_h}}_{\text{zusätzliche Masse}} = l \left( \frac{y \cdot v_h \cdot w}{p_h} \right)^2 \tag{4.7}$$

**Tabelle 4.5:** Für die Beschreibung der Überladeplanung eingeführte Variablen.

| Symbol | Beschreibung |
|---|---|
| $l_i$ | Korntankfüllstand $\geq 0$ des Mähdreschers bei Knoten $i$ |
| $u_i$ | Gesamtmasse $\geq 0$, die bis zu Knoten $i$ in das Überladefahrzeug überladen wurde |
| $e_j$ | Füllstand $\geq 0$ des Überladefahrzeugs nach dem $j$-ten Überladevorgang |
| $x_j$ | Index $\in \{0, \ldots, k\}$ in der Knotenreihenfolge, bei dem der $j$-te Überladevorgang gestartet wird |

und so weiter. Die tatsächlich zu überladene Menge berechnet sich folglich aus dem Korntankfüllstand $l$ beim Starten des Überladevorgangs und der Reihe $\alpha$

$$\underbrace{l \sum_{j=0}^{\infty} \left( \frac{y \cdot v_h \cdot w}{p_h} \right)^j}_{\alpha} = \alpha l. \tag{4.8}$$

Da der Summenterm mit zunehmender Potenz schnell kleiner wird, da wesentlich schneller überladen als Erntegut aufgenommen wird ($p_h \gg y \cdot v_h \cdot w$), reicht die Betrachtung der Reihe bis $j = 3$

$$\alpha \approx 1 + \frac{y \cdot v_h \cdot w}{p_h} + \left( \frac{y \cdot v_h \cdot w}{p_h} \right)^2 + \left( \frac{y \cdot v_h \cdot w}{p_h} \right)^3. \tag{4.9}$$

Der Korntankfüllstand $l_i$ des Mähdreschers an einem Knoten $i$ berechnet sich aus der Differenz der insgesamt bis zu diesem Knoten aufgenommenen ($m_i$) und überladenen ($u_i$) Masse als

$$l_i = m_i - u_i. \tag{4.10}$$

Die bis zu einem Knoten abgetankte Masse $u_i$ ist abhängig von der Wahl der Startpunkte $x_j$ der Überladevorgänge und berechnet sich als

$$u_i = \sum_{j=0}^{|x|-1} \begin{cases} p_h(t_i - t_{x_j}), & \text{falls } t_i > t_{x_j} \text{ und } p_h(t_i - t_{x_j}) < \alpha l_{x_j} \\ \alpha l_{x_j}, & \text{falls } p_h(t_i - t_{x_j}) \geq \alpha l_{x_j} \\ 0 & \text{sonst.} \end{cases} \tag{4.11}$$

In Formel 4.11 werden alle geplanten Überladeprozesse betrachtet. Wenn der $j$-te Überladeprozess zu einem späteren Zeitpunkt startet als Knoten $i$ erreicht wird ($t_{x_j} > t_i$), so geht dieser nicht in die Summe ein. Ist der $j$-te Überladeprozess zum Zeitpunkt $t_i$ vollständig abgeschlossen ($p_h(t_i - t_{x_j}) \geq \alpha l_{x_j}$), so geht die gesamte Überlademenge $\alpha l_{x_j}$ des Vorgangs in die Summe ein. Ist der $j$-te Überladeprozess zum Zeitpunkt $t_i$ begonnen ($t_i > t_{x_j}$), aber noch nicht abgeschlossen ($p_h(t_i - t_{x_j}) < \alpha l_{x_j}$), so geht die entladene Teilmenge $p_h(t_i - t_{x_j})$ in die Summe ein.

Die Folge von Knoten $x_0, x_1, \ldots, x_{|x|-1}$, an denen Überladeprozesse starten, muss streng monoton steigen

$$x_i < x_j, \ \forall i, j \in \{0, 1, \ldots, |x| - 1\}, \ i < j \tag{4.12}$$

Der Füllstand des Mähdreschers darf nie die Kapazität seines Korntanks überschreiten. Aus dieser Forderung ergibt sich die Bedingung

$$l_i \leq c_h, \forall i \in \{0, 1, \ldots, k-1\}. \tag{4.13}$$

Da der Füllstand monoton bis zu den Abtankorten steigt, ist es ausreichend, den Füllstand an diesen Maxima am Ende der Knotenreihenfolge zu überprüfen

$$l_i \leq c_h, \forall i \in \{x, k-1\}. \tag{4.14}$$

Die vom Überladefahrzeug abgeladene Masse $e_j$ nach dem $j$-ten Überladevorgang berechnet sich als

$$e_j = \begin{cases} e_{j-1} + \alpha l_{x_j}, & \text{falls } e_{j-1} + \alpha l_{x_j} \leq c_u \\ \alpha l_{x_j} & \text{sonst,} \end{cases} \quad \forall j \in \{1, 2, \ldots, |x|-1\} \tag{4.15}$$

mit

$$e_0 = \alpha l_{x_0}. \tag{4.16}$$

Zwischen den Überladeprozeduren muss mindestens die Zeitspanne $\epsilon$ liegen. Falls die abzutankende Masse nicht mehr in den Tank des Überladefahrzeugs passt, muss das Überladefahrzeug vorher entleert werden. Dafür muss mindestens die dafür vorgesehene Zeitspanne $\delta > \epsilon$ zwischen den Überladefenstern liegen.

$$t_{x_i} - t_{x_i-1} \geq \begin{cases} \delta, & \text{falls } e_{i-1} + \alpha l_{x_i} \leq c_u \\ \epsilon & \text{sonst.} \end{cases} \quad \forall i \in \{2, 3, \ldots, \beta\} \tag{4.17}$$

Alle Knoten $i$, die während der Dauer des Überladens besucht werden, müssen dafür geometrisch geeignet sein und die Fläche für das Überladefahrzeug muss frei sein ($valid_i = 1$)

$$\sum_{j=0}^{|x|-1} \sum_{i=x_i}^{t_i < t_{x_j} + \frac{\alpha l_{x_j}}{p_h}} (1 - valid_i) = 0 \tag{4.18}$$

und die Knoten müssen auf derselben Fahrspur liegen

$$\sum_{j=0}^{|x|-1} \sum_{i=x_i}^{t_i < t_{x_j} + \frac{\alpha l_{x_j}}{p_h}} |track_{x_j} - track_i| = 0 \tag{4.19}$$

Die Aufgabe der Überladeplanung ist es, unter Berücksichtigung der Einschränkungen aus Formeln (4.11), (4.12), (4.14), (4.18) und (4.19) eine Menge an Überladepunkten zu finden, die eine Funktion $f$ optimiert.

$$\text{minimiere } f(x) = \sum_{j=0}^{|x|-1} |l_{x_i} - \delta c_h| \tag{4.20}$$

Für die Überladeplanung wurde die Funktion $f$ aus Formel (4.20) genutzt. Diese summiert für alle Überladevorgänge den Differenzbetrag zwischen tatsächlich überladenener und vorgegebener Überlademenge. Vor Prozessbeginn kann der prozentuale Füllstand $\delta$, bei dem möglichst immer überladen werden soll, in der Planungskonfiguration vorgegeben werden. Es sind viele andere Definitionen für eine Optimierungsfunktion des Überladens denkbar. So könnte beispielsweise die Aufsummierung des Differenzbetrags der Kapazität des Mähdreschers und der Überlademenge für alle Überladevorgänge zur Minimierung der Anzahl an Überladevorgängen genutzt werden. Hier wurde die die Funktion $f$ aus Formel (4.20) genutzt, da so mit dem Parameter $\delta$ die Stabilität des Prozesses gesteuert werden kann. Desto geringer der Wert für $\delta$, desto früher und öfter wird überladen, je höher der Wert, desto später und weniger oft wird überladen.

Die Anzahl aller theoretisch möglichen Kombinationen für $x$ bei $k$ Knoten beträgt $2^k$. Durch die Einschränkung der strengen Monotonie in $x$ aus Formel (4.12) kann eine Reihenfolge ausgedrückt werden durch einen binären Vektor der Länge $k$. Der binäre Wert der $i$-ten Stelle sagt dabei aus, ob Knoten $i$ in $x$ enthalten ist und ob somit dort ein Überladeprozess starten soll. Ohne Betrachtung der Einschränkungen sind entsprechend theoretisch alle $2^k$ Werte für diesen binären Vektor möglich. □

Da sehr viele dieser theoretisch möglichen Kombinationen gegen die beschriebenen Bedingungen verstoßen, bieten sich Verfahren zur Lösung an, die ungültige Unterräume des Suchraums möglichst frühzeitig ausklammern.

### 4.4.4 Abstrakte Betrachtung der Planung von Überladeprozessen

In diesem Abschnitt wird die Aufgabe der Planung von Überladeprozessen von den Spezifika des Infield-Transportlogistik Szenarios losgelöst und für eine Komplexitätsabschätzung mit vereinfachenden Annahmen dargestellt.

Gegeben sei eine Flotte mit einer konstanten Anzahl $k$ an Mähdreschern, die instantan abtankbar sind und beim Überladen immer vollständig entleert werden. Weiterhin seien die Ladekapazitäten der Fahrzeuge als Ressourcen und das Aufnehmen von Erntegut als Verbrauch von Ressourcen (Freiraum im Korntank) definiert. So kann die Überladeplanung für ein Überladefahrzeug und mehrere Mähdrescher folgendermaßen abstrakt beschrieben werden:

Eine Gruppe an $k$ mobilen Arbeitsmaschinen mit der Menge $R$ einer Ressource soll jeweils $n$ Arbeitsschritte durchführen. Bei jedem Arbeitsschritt wird eine variierende, a priori bekannte Menge dieser Ressource verbraucht. Die Maschinen kommen eine beschränkte Anzahl an Arbeitsschritten mit dieser Menge der Ressource aus. Diese Ressourcen müssen daher von einem Lieferanten aufgestockt werden, der zwischen den Maschinen und einem Depot mit unbegrenzter Menge der Ressource pendelt. Jeder Arbeitsschritt $w$ der Maschinen benötigt eine Zeit $t_w > 0$ und verbraucht eine definierte Menge $r_w \geq 0$ der Ressource. Dabei sind $k$, $r$, $t$ und $R$ konstant, während $n$ eine variable Eingabegröße ist. Eine Maschine wird aufgestockt auf die Gesamtmenge $R$ der Ressource, wenn sie von dem Lieferanten bedient wird. Die benötigte Fahrzeit zu einer Maschine oder zum Depot ist abhängig von den aktuellen Maschinenpositionen und variiert somit. Diese Zeiten seien a priori bekannt und in der Tabelle $d$ gespeichert. Ist dann die folgende

Aufgabenstellung in polynomieller Laufzeit lösbar?

Gegeben

$n \in \mathbb{N}$: Anzahl der Arbeitsschritte pro Maschine

$k \in \mathbb{N}$: Anzahl an mobilen Arbeitsmaschinen

$r \in \mathbb{R}^{+k \times n}$: Benötigte Ressourcenmenge $r_{ij}$ für Arbeitsschritt $j$ von Maschine $i$

$t \in \mathbb{R}^{+k \times n}$: Benötigte Zeit $t_{ij}$ für Arbeitsschritt $j$ von Maschine $i$

$d \in \mathbb{R}^{+k(n+1) \times k(n+1)}$: Fahrzeiten $d_{ab}$ zwischen Positionen $a$ und $b$ für den Lieferanten

finde eine zulässige Reihenfolge für den Lieferanten, so dass für jeden Arbeitsschritt aller Maschinen immer eine ausreichende Menge der Ressource vorhanden ist.

Eine Lösung[1] für diese Aufgabenstellung ist die Konstruktion eines Hilfsgraphen und die Anwendung eines *Pareto Shortest Path* Algorithmus. Dieser Algorithmus findet den multikriteriell kürzesten Pfad [62] von einem Start- zu einem Zielknoten. Dazu werden alle $k \cdot n$ Arbeitsschritte durch Knoten repräsentiert. Für jeden Arbeitsschritt wird anhand von $d$ und $t$ überprüft, welche Arbeitsschritte rechtzeitig erreicht werden können: Es wird dem Graphen immer dann eine gerichtete Kante hinzugefügt, wenn der Lieferant in zulässiger Zeit von einer Maschinenposition eines Arbeitsschritts zum Depot und anschließend zur Maschinenposition des anderen Arbeitsschritts gelangen kann. Zusätzlich werden ein Start- und ein Endknoten eingefügt, zwischen denen ein Pfad gesucht werden soll. An jedem erreichbaren Knoten werden mehrere Vektoren $\vec{c}$ mit jeweils $k$ Einträgen gespeichert. In jedem solchen Vektor wird an der $k$-ten Stelle die Information gespeichert, an welchem Knoten die $k$-te Arbeitsmaschine zuletzt beliefert wurde. An dem Startknoten wird initial ein Nullvektor der Länge $k$ gespeichert. Nun wird nach dem in Algorithmus 2 beschriebenen Vorgehen ein Pfad vom Start- zum Zielknoten gesucht.

In dem Hilfsgraphen $G = (V, E)$ wird bei diesem Vorgehen jedes Paar an Knoten und Vektoren genau einmal betrachtet. Es gibt $k \cdot n + 2$ Knoten und jeweils maximal $n^k$ viele verschiedene Vektoren. Für jede Kante muss überprüft werden, ob sie mit dem Vektor $\vec{c}$ konsistent ist. Bei diesem Schritt wird bestimmt, ob zwischen den Lieferungen zu viel Zeit vergangen ist. Der Schritt benötigt die Zeit $\mathcal{O}(n)$ pro Kantenrelaxierung. Die Laufzeit beträgt bei $m$ Kanten demnach insgesamt

$$\mathcal{O}(k \cdot n^{k+1} + m \cdot n^{k+1}).$$

Da $m = \mathcal{O}(n^2)$ und $k < n$ angenommen werden kann, beträgt die Laufzeit im schlechtesten Fall

$$\mathcal{O}(n^{k+3}).$$

Es existiert somit eine polynomielle Lösung für die abstrakte und vereinfachte Formulierung der Überladeplanung. Eine genaue Analyse der Komplexität für die in Abschnitt 4.4.3 beschriebene Problemstellung der Überladeplanung steht aus.

Der gezeigte Ansatz liefert eine zulässige Lösung. Für das Auffinden der optimalen Lösung müsste er entsprechend angepasst werden. Durch die Online-Problematik und die beschränkte

[1] Vielen Dank für den Vorschlag dieser Lösung an Prof. Dr. rer. nat. Markus Chimani.

**Algorithmus 2** Pareto Shortest Path für das Finden einer zulässigen Reihenfolge an Belieferungen von mobilen Arbeitsmaschinen in einem Hilfsgraphen $G$ mit eingefügten Knoten für den Start $s$ und das Ende $z$.

```
1: Prozedur ParetoShortestPath(G, s, z)
2:     solange es existiert ein Knoten mit unbetrachtetem Vektor führe aus
3:         wähle einen Knoten v mit einem solchen Vektor c⃗
4:         für alle Kanten e von v nach u führe aus
5:             falls ausgehend von c⃗ der Übergang von v nach u gültig ist dann
6:                 c⃗' ← c⃗
7:                 c'_i ← v, mit i: Nummer der Arbeitsmaschine des Knotens v
8:                 falls c⃗' nicht schon für u vorhanden dann
9:                     füge c⃗' dem Knoten u hinzu
10:                Ende falls
11:            Ende falls
12:        Ende für
13:    Ende solange
14: Ende Prozedur
```

Rechenkapazität der Fahrzeugrechner werden an die Laufzeit der Überladeplanung hohe Anforderungen gestellt. Daher ist es gut möglich, dass der gezeigte Ansatz, falls er übertragbar auf die genannte Problemstellung ist, trotz der polynomiellen Laufzeit nicht für einen Einsatz in der Praxis geeignet ist. Deshalb wird für die Berechnung der Überladeplanung das in Abschnitt 4.4.5 beschriebene Verfahren einer impliziten Baumsuche genutzt, das auch ohne erschöpfende Suche gültige Lösungen generieren kann. Denn eine gültige und in annehmbarer Laufzeit ermittelte, potentiell suboptimale Lösung trägt im Gegensatz zu einer optimalen Lösung, die während der Prozessausführung nicht ermittelt werden kann, der Effizienzsteigerung eines realen Prozesses bei.

### 4.4.5 Berechnung der Überladeprozesse mit impliziter Baumsuche

Wie in Abschnitt 4.4.3 beschrieben, gibt es theoretisch $2^k$ mögliche Überladefolgen für Knotenreihenfolgen des Mähdreschers mit $k$ Elementen. Aufgrund der Größe des Suchraums kommt für die Überladeplanung in annehmbarer Zeit auf einem Fahrzeugrechner demnach keine erschöpfende Suche in Frage. Da jedoch eine beliebige Folge von Überladeprozessen ungültig ist, sobald einer der enthaltenen Überladeprozesse ungültig ist, können große Teile des Suchbaums ausgeklammert werden. Dazu bieten sich einige gängige Verfahren an. Zwei Beispiele sind der A* [40] und der Verzweigen-und-Abschneiden (*Branch-and-Bound*) [53] Algorithmus.

Bei dem A* Algorithmus handelt es sich um ein informiertes Suchverfahren nach der ersten besten Lösung von einem Start- zu einem Zielknoten in einem Graphen. Neben einer Kostenfunktion gibt es eine Heuristikfunktion, die eine Abschätzung der verbleibenden Kosten bis zu einem Zielknoten liefert. Eine solche Heuristik ist zulässig, wenn sie die Kosten zum Ziel nie überschätzt. Für zulässige Heuristiken ist der Algorithmus vollständig und sogar optimal effizi-

ent. Das bedeutet, der Algorithmus findet in diesem Fall immer die beste Lösung und expandiert dabei die minimale Anzahl an Knoten.

In dieser Arbeit wird für die Planung von Überladeprozessen ein A* ähnliches Verfahren in einer impliziten Baumsuche eingesetzt. Ungültige Teillösungen werden bei diesem Verfahren nicht expandiert. Bei dem Expandieren von Knoten werden anhand einer Kosten- und Heuristikfunktion vielversprechende Lösungen zuerst expandiert. Anders als bei dem A* Algorithmus kann ein Knoten nicht über unterschiedliche Vorgängerknoten erreicht werden, da die Knotenhistorie eindeutig die vorhergehenden Überladeprozesse kodiert.

Die Knoten des Baumes haben die Attribute

- **Identifikationsnummer** des Knotens (0 für den Wurzelknoten)
- **Vorgänger** des Knotens in dem Suchbaum
- **Index** des Knotens in der Fahrspurreihenfolge des Mähdreschers (-1 für Überladeprozedur des Überladefahrzeugs, $\geq 0$ sonst)
- **Kosten** des Knotens
- **Heuristik** des Knotens.

In den Erntetests wurden die Kosten- und Heuristikfunktion füllstandsbasiert berechnet. Die Kosten eines Knotens berechnen sich aus den Kosten des Vorgängerknotens und der Differenz der angepeilten und tatsächlich überladenen Menge (siehe Formel 4.20) an Erntegut als

$$c(v) = \underbrace{c(pred(v))}_{\text{Kosten des Vorgängerknotens}} + \underbrace{|m_{ziel} - m_{überladen}|}_{\text{Differenz angepeilte und tatsächliche Überlademenge}} . \tag{4.21}$$

Als Heuristik wird die verbleibende Anzahl an Überladevorgängen geschätzt und mit einer für die Schätzung angenommenen Abweichung von überladener und angepeilter Erntegutmasse multipliziert. Dazu wird die nach einem Überladevorgang verbleibende Erntemenge $m_h$ im Korntank des Mähdreschers und die noch abzuerntende Getreidemenge des Feldes $e_v$ durch den Schwellwert $m_{max}$ für das Starten eines Überladevorgangs geteilt. Dieser Wert wird mit der Abweichung des Schwellwerts von der angepeilten Überlademenge multipliziert.

$$h(v) = \underbrace{\frac{m_h + e_v}{m_{max}}}_{\text{geschätzte Anzahl Überladevorgänge}} \cdot \underbrace{|m_{max} - m_{ziel}|}_{\text{Abweichung Schwellwert von angepeilter Überlademenge}} \tag{4.22}$$

In Algorithmus 3 ist der Ablauf der impliziten Baumsuche zusammengefasst. Ausgehend von den in Abschnitt 4.4.3 beschriebenen Eingabewerten (siehe Tabelle 4.1 und Tabelle 4.4) wird ein gültiger und nach der Kostenfunktion aus (4.20) optimaler Überladeplan gesucht. Dazu wird zunächst eine leere Baumstruktur, eine offene Knotenliste und eine leere Zielknotenliste angelegt. In die Baumstruktur und die offene Knotenliste wird dann der Wurzelknoten hinzugefügt, bevor die Schleife zur Baumerstellung ausgeführt wird. Diese Schleife wird so lange ausgeführt, bis entweder kein Knoten mehr in der offenen Knotenliste vorhanden ist, oder bis eine maximale

Ausführungszeit erreicht wurde. In dem ersten Fall wären gültige Teile des Baums komplett abgesucht worden, im zweiten Fall nur ein vielversprechender Teil. Da der Suchraum exponentiell mit der Länge der Knotenreihenfolge des Mähdreschers steigt und die Ausführzeit begrenzt ist, tritt in der Regel der zweite Fall ein. In der Schleife wird jeweils der nach der Kosten- und Heuristikfunktion beste Knoten ausgewählt und aus der offenen Knotenliste entfernt. Falls der Knoten ein gültiger Zielknoten ist, so wird er der Zielknotenliste hinzugefügt. Ein Knoten ist genau dann ein gültiger Zielknoten, falls der Überladeplan für jeden Knoten in der Fahrspurreihenfolge des Mähdreschers, also für den gesamten verbleibenden Prozess, keine der in Abschnitt 4.4.3 beschriebenen Bedingungen verletzt. Ist der Knoten kein gültiger Zielknoten, so wird dieser nach Algorithmus 4 expandiert.

**Algorithmus 3** Überladeplanung mit Baumsuche für den Routengraphen $G$ und eine Probleminstanz $P$. Die Probleminstanz beinhaltet unter anderem aktuelle Füllstände und Positionen der Fahrzeuge und ermöglicht eine dynamische Planung.

```
1: Prozedur BaumsucheÜberladen(G, P)
2:     initialisiere leere Baumstruktur T
3:     initialisiere leere offene Knotenliste O
4:     initialisiere leere Zielknotenliste R
5:     füge Wurzelknoten s zu T und O hinzu
6:     solange O hat Einträge und Ausführzeit nicht überschritten führe aus
7:         wähle besten Knoten u aus O anhand von c(u) + h(u)
8:         entferne u aus O
9:         falls u ist gültiger Zielknoten dann
10:            füge u zu R hinzu
11:        sonst
12:            Expandiere(u, T, G, P)
13:        Ende falls
14:    Ende solange
15:    falls R hat Einträge dann
16:        bestimme besten Zielknoten u aus R anhand von f(u)
17:        rekonstruiere Liste x an Überladeprozessen für u anhand seiner Vorgänger
18:        gib zurück x
19:    sonst
20:        gib zurück leere Liste
21:    Ende falls
22: Ende Prozedur
```

Falls der zu expandierende Knoten der Wurzelknoten ist, so wird bei der Expansion mit dem ersten Knoten der Knotenreihenfolge angefangen. Andernfalls werden Knoten ab dem Index in der Fahrspurreihenfolge des Mähdreschers betrachtet, der die Mähdrescherposition nach dem vorhergehenden Überladeprozess beschreibt. Solange der Korntank des Mähdreschers nicht den maximal zulässigen Füllstand $\beta \cdot c_h$ überschreitet, werden in einer Schleife die folgenden Knoten darauf hin überprüft, ob sie ein gültiger Startpunkt für einen Überladeprozess sind. Bei dieser Überprüfung werden alle Bedingungen aus Abschnitt 4.4.3 berücksichtigt. Ist ein Knoten gültig,

so wird er der offenen Knotenliste hinzugefügt. Hat der in einer Schleife betrachtete Knoten keinen Nachfolger in der Knotenreihenfolge des Mähdreschers, so wird die Schleife unterbrochen.

**Algorithmus 4** Expansion eines Knotens $u$ des Suchbaums $T$ für einen Routengraphen $G$ und eine Probleminstanz $P$.

```
1: Prozedur EXPANDIERE(u, T, G, P)
2:     i ← 0
3:     falls u ist nicht Wurzelknoten von T dann
4:         i ← Knotenindex der Mähdrescherposition nach vorigem Überladen
5:     Ende falls
6:     solange Füllstand bei Knoten i zulässig  führe aus
7:         falls v_i ist gültiger Start für Überladeprozess dann
8:             berechne Knotenattribute für neuen Knoten
9:             füge neuen Knoten zu T hinzu
10:        Ende falls
11:        falls i hat keinen Nachfolger in der Knotenreihenfolge dann
12:            unterbreche
13:        Ende falls
14:        i ← i + 1
15:    Ende solange
16: Ende Prozedur
```

Mit der Expansion des Knotens ist der Schleifendurchlauf abgeschlossen und der nächste Durchlauf beginnt. Nachdem die Schleife aufgrund einer der beiden genannten Kriterien verlassen wird, wird die Zielknotenliste untersucht. Ist diese leer, so konnte kein gültiger Überladeplan gefunden werden. Andernfalls werden die Überladepläne anhand der Vorgänger der Knoten rekonstruiert und mit durch die Funktion $f$ bewertet. Die am Besten bewertete Reihenfolge wird abschließend zurückgegeben.

Die Baumsuche ist nicht nur für die initiale oder statische Vorplanung von Überladeprozessen, sondern auch für deren dynamische Neuplanung während der Prozessausführung geeignet. Die Knotenreihenfolge des Mähdreschers und die Attribute der enthaltenen Knoten (Tabelle 4.4) müssen für eine Neuplaung neu berechnet und der Baumsuche gemeinsam mit den aktuellen Fahrzeugfüllständen und -positionen übergeben werden. So kann das Verfahren zu einem beliebigen Zeitpunkt eine Neuplanung durchführen.

## 4.5 Zusammenfassung der Routenplanung

In diesem Kapitel wurde die maschinenübergreifende Routenplanung für ein Ernte- und ein Transportfahrzeug in dem Szenario der Infield-Transportlogistik aus Abschnitt 2.2 beschrieben. Wie in Abschnitt 4.2 erläutert, wird die Planung des gesamten Prozesses in zwei Schritte unterteilt. In dem ersten Schritt wird die Fahrspurreihenfolge des Erntefahrzeugs nach Beetstrategien aus der landwirtschaftlichen Praxis (Abschnitt 4.3) festgelegt, während die Überladeplanung in einer impliziten Baumsuche anhand einer Kosten- und Heuristikfunktion geschieht (Abschnitt 4.4).

Die Ergebnisse der beiden Planungsschritte werden zu konsistenten Routenplänen für beide Fahrzeuge zusammengefasst. Wie in Abschnitt 3.2.1 als technische Randbedingung beschrieben, wird zwischen den Fahrzeugen Funkkommunikation eingesetzt, bei der mit schwankender Bandbreite zu rechnen ist. Daher werden die Routenpläne jeweils durch einen minimalen Satz an Routenpunkten beschrieben, um die Pläne effizient kommunizieren zu können. Die Route des Mähdreschers besteht entsprechend jeweils aus dem Start- und dem Endpunkt für jede abzuarbeitende Fahrspur und aus einem Start- und Endpunkt für jeden Überladevorgang. Die Route des Überladefahrzeugs besteht jeweils aus einem Start- und einem Endpunkt für Überladevorgänge des Mähdreschers und für das Abladen am Feldrand. Während der Prozessausführung können die Routenpläne bei Abweichungen in dem vorgestellten Verfahren zur Überladeplanung nach Bedarf dynamisch neu geplant werden. Abbildung 4.17 zeigt exemplarisch das Ergebnis der Routenplanung für eine Feldgeometrie mit Prozess- und Maschinenkonfiguration aus realen Erntetests des Gesamtsystems, die in Abschnitt 6.3 beschrieben werden. Diese Routen werden an die Fahrzeuge verteilt und können auf diesen durch die Bewegungsplanung zu ausführbaren Plänen verfeinert werden (siehe Abschnitt 3.2.3).

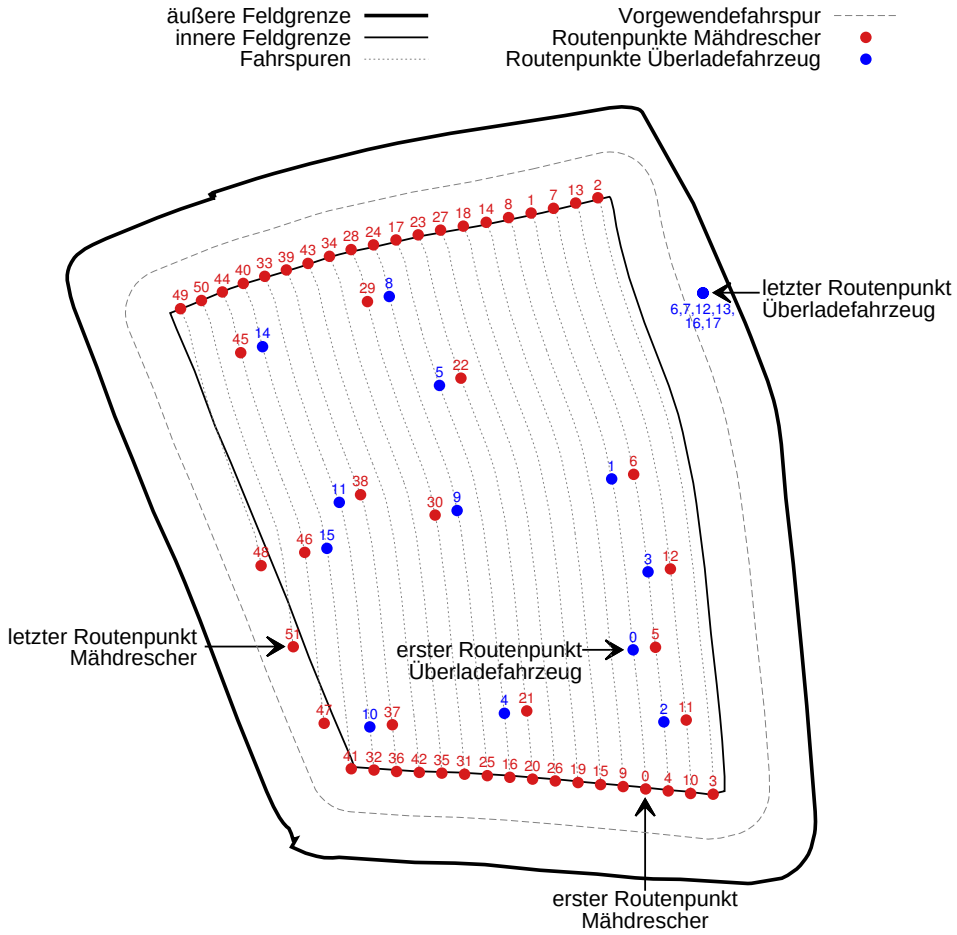



**Abbildung 4.17:** Beispielergebnis der Routenplanung für eine Felgeometrie mit Prozess- und Maschinenkonfigurationen aus Erntetests mit realen Maschinen. Die Routenpunkte des Mähdreschers sind rot, die des Überladefahrzeugs blau dargestellt. Die Routenpunkte sind jeweils durchnummeriert. Der erste Überladevorgang ist beispielsweise durch die Routenpunkte 0 (Start) und 1 (Ende) des Überladefahrzeugs und die Routenpunkte 5 (Start) und 6 (Ende) des Mähdreschers beschrieben.

## 4.6 Fazit zur Routenplanung

Dieser Abschnitt diskutiert die Erreichung der in Abschnitt 1.1 definierten Zielsetzungen dieser Arbeit bezüglich der maschinenübergreifenden Routenplanung. In dem vorgestellten Ansatz aus Abschnitt 4.2 wird diese in zwei nachgeordnete Planungsschritte aufgeteilt. Die entstehenden Planungsaufgaben von Fahrspurreihenfolgen und Überladeprozessen wurden detailliert dargestellt, so dass auf dieser Basis Planungsverfahren implementiert werden konnten. Die Zielsetzung der Formalisierung wurde somit erreicht. Mit dem in Abschnitt 4.4.5 beschriebenen Verfahren zur Überladeplanung wurde außerdem die Zielsetzung der raumzeitlichen Planung kooperativer Prozessabschnitte der beteiligten Maschinen erreicht.

Durch den zentralen Ansatz der maschinenübergreifenden Planung gibt es keine konkurrierenden

Routenpläne. Alle Teilpläne, die als Routen an die Fahrzeuge verteilt werden, beschreiben immer denselben gemeinsamen Plan eines Gesamtprozesses. Die Routenpläne werden nur bei vorhandener Funkkommunikation erstellt und verteilt. Die Zielsetzung der Konsistenz von erstellten Routenplänen wurde daher ebenfalls erreicht.

Eine weitere wichtige Zielsetzung der Arbeit ist die Prozessoptimierung. In Abschnitt 4.2 wurde bereits beschrieben, dass die Ernte- und die Transportplanung zwei Optimierungsprobleme mit potentiell gegenläufigen Zielen darstellen, die sich gegenseitig raumzeitlich beeinflussen. Außerdem kommt erschwerend hinzu, dass es sich aufgrund von hoher Prozessdynamik um ein Online-Problem handelt (siehe Abschnitt 4.2). Eine erschöpfende Suche ist für das Auffinden des Optimums ungeeignet, da für jeden möglichen Ernteplan jeder möglicher Transportplan untersucht werden müsste. Für $n$ Fahrspuren gibt es $2 \cdot n!$ mögliche Fahrspurreihenfolgen, die jeweils eine unterschiedliche Abarbeitung des Feldes repräsentieren. Jede der $n!$ Permutation der Fahrspurnummern $\{0, \ldots, n-1\}$ repräsentiert eine Fahrspurreihnfolge, für die es es zwei unterschiedliche Fahrtrichtungen gibt. Für jede dieser $2 \cdot n!$ möglichen Fahrspurreihenfolgen gibt es wiederum eine Vielzahl an möglichen Überladeplänen. Wie viele Überladepläne es gibt, hängt in dem vorgestellten Verfahren von der Auflösung der Diskretisierung und der Größe der Probleminstanz ab. Für eine Anzahl an $k$ Knoten entlang der Fahrspurreihenfolge des Mähdreschers gibt es, wie in Abschnitt 4.4.3 erläutert theoretisch $2^k$ mögliche Lösungskandidaten. Von diesen sind jedoch viele ungültig und können frühzeitig ausgeschlossen werden. Trotzdem wird deutlich, dass die Verschachtelung dieser Planungsschritte in einer hohen Komplexität resultiert. Zusätzlich erschweren die gegenseitige raumzeitliche Beeinflussung der Optimierungsprobleme und die prozessspezifischen Einschränkungen die Planung. So ist es durchaus möglich, dass ein Optimierungsschritt aus Sicht der Fahrspurreihenfolge zu keinem gültigen Gesamtplan führt, wenn der Ernteprozess aufgrund von Wartezeiten unterbrochen werden muss. Für die gesamte Routenplanung ist zu vermuten, dass es sich um ein NP-schweres Planungsproblem handelt, das nicht effizient lösbar ist. Eine dementsprechende Komplexitätsbetrachtung steht jedoch aus.

Da es sich um ein Online-Problem handelt, wurde die Planung wie beschrieben in weniger komplexe und nachgeordnete Aufgaben unterteilt. Der erste Planungsschritt der Festlegung einer Fahrspurreihenfolge geschieht anhand von fixen Beetstrategien aus der landwirtschaftlichen Praxis. Hier findet also keine Optimierung statt. Der zweite Teil ist die Planung der Überladeprozesse. Bei dieser Überladeplanung wird eine implizite Baumsuche anhand einer Kosten- und Heuristikfunktion entlang einer festgelegten Fahrspurreihenfolge des Mähdreschers durchgeführt. Für den Fall einer zulässigen Heuristik ist diese Suche vollständig. Das bedeutet, in diesem Fall findet die Suche bei entsprechender Laufzeit immer die beste Lösung, sofern eine Lösung vorhanden ist. Hier wird der Prozess hinsichtlich der vorgegebenen Funktion[2] aus Formel (4.20) optimiert, wobei für die jeweilige Fahrspurreihenfolge das Optimum erreicht werden kann. In der aus diesen beiden Planungsschritten bestehenden Routenplanung wird somit eine Optimierung durchgeführt, die allerdings nicht das globale Optimum für das Infield-Transportlogistik Szenario finden kann, da die Abarbeitung des Feldes durch den Mähdrescher nach einer vorgegebenen Strategie geschieht.

[2]Die beschriebene Optimierungsfunktion könnte zukünftig durch komplexere Funktionen ersetzt werden, etwa um verschiedene Kostenfaktoren gegeneinander aufzurechnen.

Abgesehen davon, ob ein globales Optimum effizient auffindbar ist, stellt sich generell die Frage, ob die Nutzung eines globalen Optimums mit beliebiger Fahrspurreihenfolge für den realen Prozess sinnvoll ist. Da es sich um ein Online-Problem handelt, muss während der Prozessausführung dynamisch neu geplant werden. Wenn nun ein globales Optimum als Plan vorgegeben wird, können kleinste Abweichungen in der Prozessausführung dazu führen, dass der Prozess suboptimal ausgeführt wird, beispielsweise weil der Mähdrescher auf das Überladefahrzeug warten muss. Die Stabilität der Prozessausführung hängt aufgrund der Online-Problematik stark davon ab, mit wie viel Sicherheit der vorgegebene Plan erstellt wurde. Das aus Prozesssicht optimale Ergebnis für ein reales Szenario ist folglich ein Kompromiss zwischen Stabilität und Optimalität eines Routenplans. Mit den eingesetzten Beetstrategien und der Optimierung der Überladevorgänge wird dies in dem vorgestellten Ansatz erreicht. Durch den Einsatz von Beetstrategien wird der Gesamtprozess robust geplant und stabil ausgeführt. Die Stabilität wird durch die Vielzahl möglicher Überladevorgänge erreicht. Zusätzlich ermöglicht die Prognose von Füllständen und Fahrzeiten sowie die vorausschauende Planung und Optimierung von Überladevorgängen eine Effizienzsteigerung von realen Prozessen, insbesondere durch das Vermeiden von Prozessunterbrechungen aufgrund von Wartezeiten. Somit wird die Zielsetzung der Prozessoptimierung erreicht.

# Kapitel 5

# Bewegungsplanung

Dieses Kapitel beschreibt die Erzeugung von abfahrbaren orts-, zeit- und geschwindigkeitsbehafteten Wegpunkten für die eingesetzten Feldfahrzeuge.

Eine Bewegungsplanung liefert im Allgemeinen aus einer Start- und einer Zielpose und einer Repräsentation der Umgebung einen abfahrbaren Pfad als Folge von Wegpunkten für eine mobile Plattform. Wie in Abschnitt 3.2.3 erläutert, werden zur Bewegungsplanung in dem beschriebenen Planungssystem für die verschiedenen Fahrzeugtypen unterschiedliche Planungsalgorithmen eingesetzt.

In Abschnitt 5.1 wird die Einordnung der Bewegungsplanung in die Architektur und die Anbindung an das Planungssystem gegeben. Abschnitt 5.2 stellt die eingesetzten Verfahren zur Bewegungsplanung vor. Die fahrzeugtypspezifische Bewegungsplanung für den Mähdrescher wird in Abschnitt 5.3, die für das Überladefahrzeug in Abschnitt 5.4 detailliert beschrieben. Experimente und Ergebnisse der Bewegungsplanung werden in Kapitel 6 erläutert.

## 5.1 Einordnung in die Architektur des Planungssystems

Im Gegensatz zu der in Kapitel 4 beschriebenen Routenplanung, die auf einer einzelnen Maschine ausgeführt wird und die für alle beteiligten Feldfahrzeuge einen gemeinsamen Routenplan erstellt, wird die Bewegungsplanung auf jedem Feldfahrzeug lokal ausgeführt. Das Ergebnis der lokalen Ausführung ist ein Bewegungsplan für das einzelne Fahrzeug, der den durch die fahrzeugübergreifende Planung vorgegebenen Routenplan in Zeit, Ort, Orientierung und Geschwindigkeit erfüllt.

### 5.1.1 Anbindung an die Routenplanung

Die am Ernteprozess beteiligten Maschinen kommunizieren per Funk. Für die Kommunikation von Planungs- und Prozessdaten kommt der in Abschnitt 3.2.4 beschriebene datenzentrische

Ansatz des sogenannten *Data Distribution Service* (DDS) zum Einsatz.

Wie in Kapitel 4 beschrieben, enthalten die Routenpläne für den Mähdrescher und das Überladefahrzeug unterschiedliche Routenpunkte. Für die Mähdrescher besteht ein Routenplan aus der Fahrspurreihenfolge und aus Überladeprozessen, das Überladefahrzeug erhält als Routenpunkte die Überladeprozesse des Mähdreschers und die Entladevorgänge am Feldrand. Ein Routenpunkt ist eindeutig definiert und enthält Vorgaben für Zeit, Ort, Orientierung, Geschwindigkeit und Füllstand einer Maschine.


**Abbildung 5.1:** Anbindung der Routen und Bewegungsplanung. Die zentrale Routenplanung stellt jedem Fahrzeug über Funk per DDS einen Routenplan bereit, der auf jeder Maschine lokal durch eine fahrzeugtypspezifische Bewegungsplanung verfeinert wird. Ergebnis der Bewegungsplanung ist ein durch die Maschine abfahrbarer Pfad, der den Routenplan erfüllt.

### 5.1.2 Anbindung an die Planausführung

Wie in Abbildung 2.1 dargestellt, wird die Planausführung der jeweiligen Maschine im Infield-Transportlogistik Szenario des Projekts *marion* durch den Projektpartner CLAAS realisiert. Die Anbindung der Planausführung an die Bewegungsplanung und somit an das Planungssystem ist in Abbildung 5.2 schematisch dargestellt. Die Bewegungsplanung stellt per *Data Distribution Service* (DDS) Wegpunkte bereit. Das Lenksystem ist per *Controller Area Network* (CAN) Datenbus auf den Fahrzeugen angebunden. Ein CAN-DDS Modul nimmt daher die von der Bewegungsplanung generierten Wegpunkte entgegen und sendet diese über den CAN Bus an das auf der Maschine installierte Lenksystem. Das Lenksystem berechnet aus den Wegpunkten ein Lenksignal, das an die Lenkung weitergegeben wird. Beim Abfahren der Wegpunkte regelt das Lenksystem die Maschinenposition auf die vorgegebene Sollstrecke.

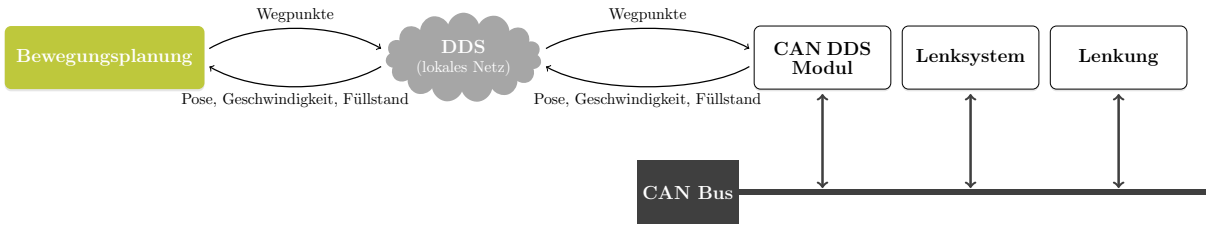



**Abbildung 5.2:** Anbindung der Planausführung an die Bewegungsplanung (angepasst von [88]). Die Bewegungsplanung stellt die berechneten Wegpunkte per DDS bereit. Ein Modul nimmt die Daten entgegen und sendet sie über den CAN Bus an das Lenksystem weiter. Das Lenksystem berechnet das erforderliche Lenksignal und regelt die Maschine mit Lenkbefehlen auf den berechneten Pfad.

Das eingesetzte Lenksystem kann maximal 49 Wegpunkte verarbeiten, die zwischen 0.5 und 1.5 Metern auseinander liegen dürfen. Der berechnete Pfad muss also immer in passende Abschnitte

unterteilt werden, wenn der Pfad aus mehr als dieser Anzahl an Wegpunkten besteht. Dabei wird vom Lenksystem eine Überlappung des alten und des neuen Pfades von 10 Metern benötigt. Das Unterteilen eines Pfades ist schematisch in Abbildung 5.3 gezeigt.

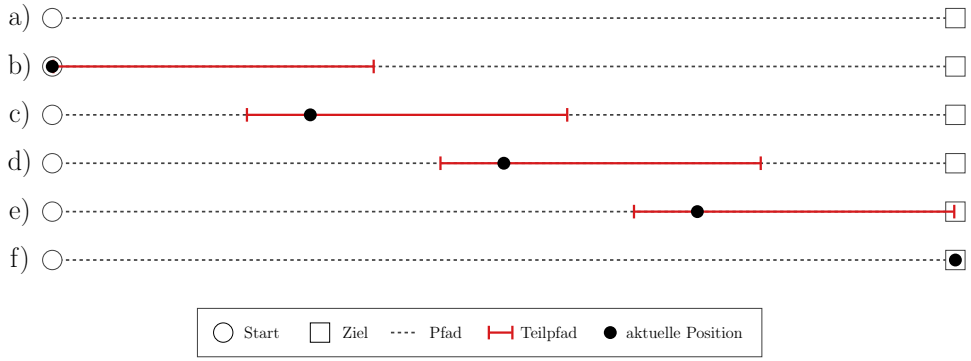



**Abbildung 5.3:** Unterteilung eines Pfades in Teilstücke. Weil das Lenksystem nur eine begrenzte Anzahl an Wegpunkten verarbeiten kann, muss der Pfad (a) zerteilt werden. Vor dem Erreichen des Endes eines Teilstücks wird ein neues Teilstück erzeugt und übermittelt (b-e). Das geschieht solange, bis das Ziel erreicht ist (f).

Da maximaler Lenkwinkel, maximale Lenkwinkeländerung pro Zeiteinheit und die Genauigkeit der Lenksysteme der Fahrzeuge a priori nicht bekannt waren, wurden entsprechende Experimente mit den eingesetzten Fahrzeugtypen durchgeführt. Aufbau und Ergebnisse der Experimente für Mähdrescher und Überladefahrzeug werden in Abschnitt 6.1 beschrieben.

### 5.1.3 Dynamische Neuplanung

Die Bewegungsplanung überwacht die Ausführung des Bewegungsplans durch die Maschinen, indem zyklisch Position und Geschwindigkeit abgefragt und mit dem Plan abgeglichen werden. Wenn die Planausführung aufgrund von Störungen oder Abweichungen in der Prozessausführung folgende Schwellwerte überschreitet, wird die Bewegungsplanung neu ausgeführt:

- räumliche Abweichung vom Plan $> 1$ m
- zeitliche Abweichung vom Plan $> 10$ s

Bei Abweichungen wird immer zuerst versucht, durch lokale Neuausführung der Bewegungsplanung den übergeordneten Routenplan einzuhalten. Sollte die Bewegungsplanung jedoch dazu nicht in der Lage sein, oder weicht der prognostizierte Füllstand bei dem nächsten Routenpunkt zu sehr von dem geplanten Füllstand ab, so muss ein neuer Routenplan für die betroffenen Fahrzeuge erstellt werden. Dazu benachrichtigt die Bewegungsplanung die Routenplanung per DDS. Von der Routenplanung wird durch dynamische Neuplanung ein neuer maschinenübergreifender Plan erstellt und den Feldfahrzeugen per DDS übertragen. Der neue Routenplan wird jeweils lokal durch die Bewegungsplanung verfeinert und durch die Maschinen ausgeführt (siehe Abbildung 5.4).

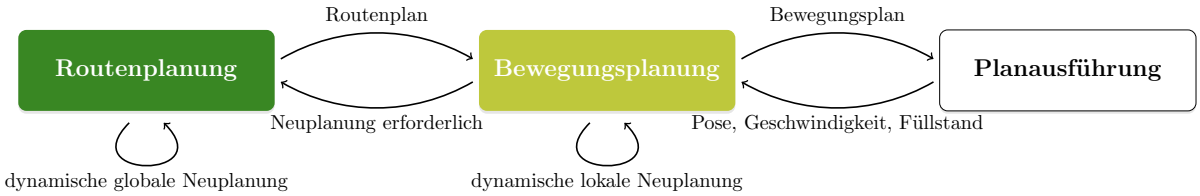
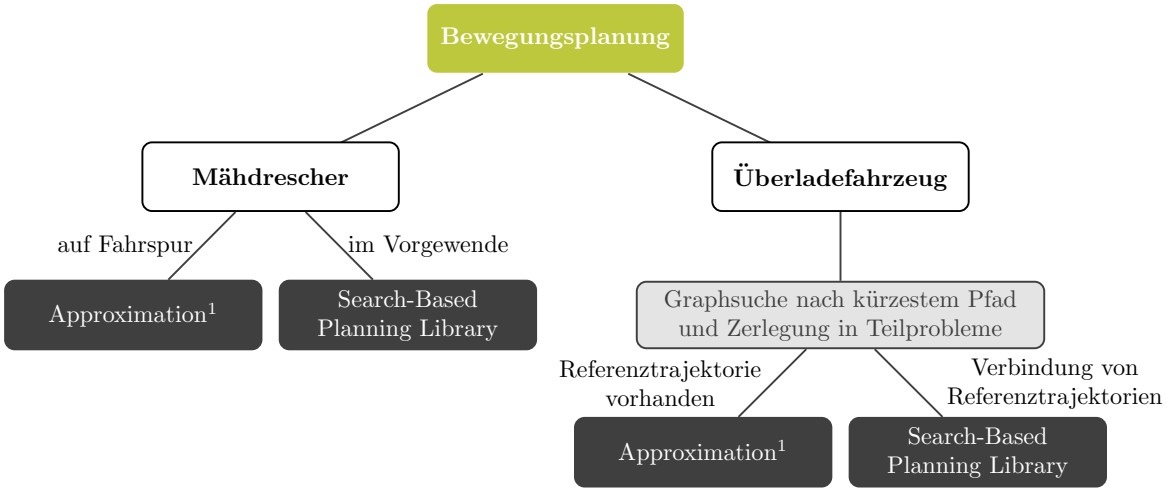



**Abbildung 5.4:** Dynamische Neuplanung bei Abweichungen in der Planausführung. Bei Abweichungen wird die Bewegungsplanung lokal neu ausgeführt. Wenn dabei der Routenplan nicht eingehalten werden kann, wird eine dynamische globale Neuplanung durch die Routenplanung durchgeführt. In diesem Fall wird der neue Routenplan auf jeder Maschine durch die Bewegungsplanung verfeinert, bevor er ausgeführt wird.

## 5.2 Eingesetzte Verfahren zur Bewegungsplanung

Für die Bewegungsplanung wird eine Kombination aus verschiedenen, in Abbildung 5.5 gezeigten Verfahren genutzt. Die Verfahren werden in den folgenden Unterkapiteln genauer beschrieben, bevor in Abschnitt 5.3 und Abschnitt 5.4 die fahrzeugtypspezifischen Vorgehensweisen für Mähdrescher und Überladefahrzeug dargestellt werden.




[1] *Approximation einer Referenztrajektorie mit Bewegungsprimitiven*

**Abbildung 5.5:** Eingesetzte Verfahren zur Bewegungsplanung für die verschiedenen Fahrzeugtypen (angepasst von [88]). Für den Mähdrescher und das Überladefahrzeug werden jeweils die *Search-Based Planning Library* (Abschnitt 5.2.1) und die *Approximation von Referenztrajektorien mit Bewegungsprimitiven* (Abschnitt 5.2.2) eingesetzt. Für das Überladefahrzeug muss zusätzlich eine Graphsuche vorgeschaltet werden, um das Kreuzen von Fahrspuren zu ermöglichen und um den noch nicht abgeernteten Teil des Feldes als dynamisches Hindernis berücksichtigen zu können.

### 5.2.1 Gitterbasierter Planer der *Search-Based Planning Library*

Die *Search-Based Planning Library* (SBPL) ist eine Bibliothek zur Bewegungsplanung nach dem Stand der Technik. Sie implementiert einen generischen Satz von Bewegungsplanern, die suchbasierte Planung nutzen [58]. Bei suchbasierter Planung wird ein Planungsproblem in einem Graphen modelliert, in dem anschließend eine Lösung gefunden werden kann. Die Bibliothek wurde von Maxim Likhachev an der University of Pennsylvania in Zusammenarbeit mit *Willow Garage* entwickelt [58]. Die SBPL ist unter anderem als Paket für das *Robot Operating System* (ROS) verfügbar [79].

Grundlage der Planung sind im Voraus berechnete, kinematisch gültige Bewegungen, sogenannte Bewegungsprimitiven, einer mobilen Plattform. Aus den Primitiven wird in einer Umgebungsrepräsentation ein Graph generiert. Anschließend kann in diesem Graphen mit üblichen Suchalgorithmen ein gültiger Pfad bestimmt werden (vergleiche Abbildung 5.6).

Die im beschriebenen Planungssystem für die Bewegungsplanung genutzte Umgebungsrepräsentation diskretisiert die Koordinaten für die Position ($x$, $y$) und für die Orientierung ($\theta$) in einem Raster. Die Zellenauflösung für $x$ und $y$ und die Winkeldiskretisierung für $\theta$ sind dabei parametrierbar und müssen mit denen der eingesetzten Bewegungsprimitiven übereinstimmen.

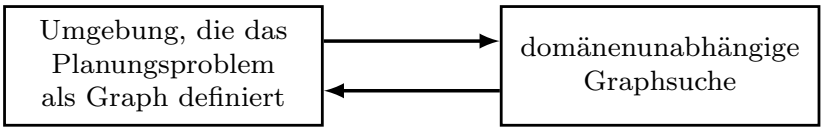



**Abbildung 5.6:** Funktionsprinzip der SBPL. Eine Umgebungsrepräsentation modelliert das Planungsproblem in einem Graphen [57]. Dieser kann anschließend mit domänenunabhängigen Suchalgorithmen untersucht werden, um einen Pfad zu bestimmen.

#### Bewegungsprimitiven

Die Bewegungsprimitiven der SBPL starten und enden immer auf dem diskreten Raster $(x, y, \theta)$. Zusätzlich hat jede Primitive noch eine beliebige Anzahl an kontinuierlichen Zwischenposen, die gemeinsam mit den Abmessungen eines mobilen Roboters für die Kollisionsvermeidung genutzt werden. Die Bibliothek verwendet standardmäßig 16 Winkelschritte, wie in Abbildung 5.7 dargestellt. Alle Primitiven starten an der diskreten Position $(0, 0)$. Zur Veranschaulichung ist in Abbildung 5.8 ein kleiner Satz an Bewegungsprimitiven für acht diskrete Winkelschritte und in Abbildung 5.9 der komplette Satz von Bewegungsprimitiven für den *PR2* Roboter von *Willow-Garage* aus der SBPL Bibliothek dargestellt.

Beim Erstellen eines Graphen werden die Primitiven, deren Startwinkel der aktuellen Orientierung entsprechen, an die aktuelle Position verschoben. Knoten mit Endpositionen, die kollisionsfreie Bewegungen darstellen, werden in den Graphen aufgenommen. Dadurch, dass der Endwinkel der vorherigen Bewegung gleich dem Startwinkel der folgenden Bewegung ist, entsteht ein Pfad mit kontinuierlicher Lenkwinkeländerung.

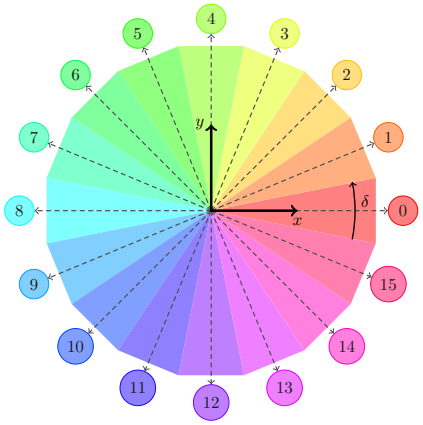
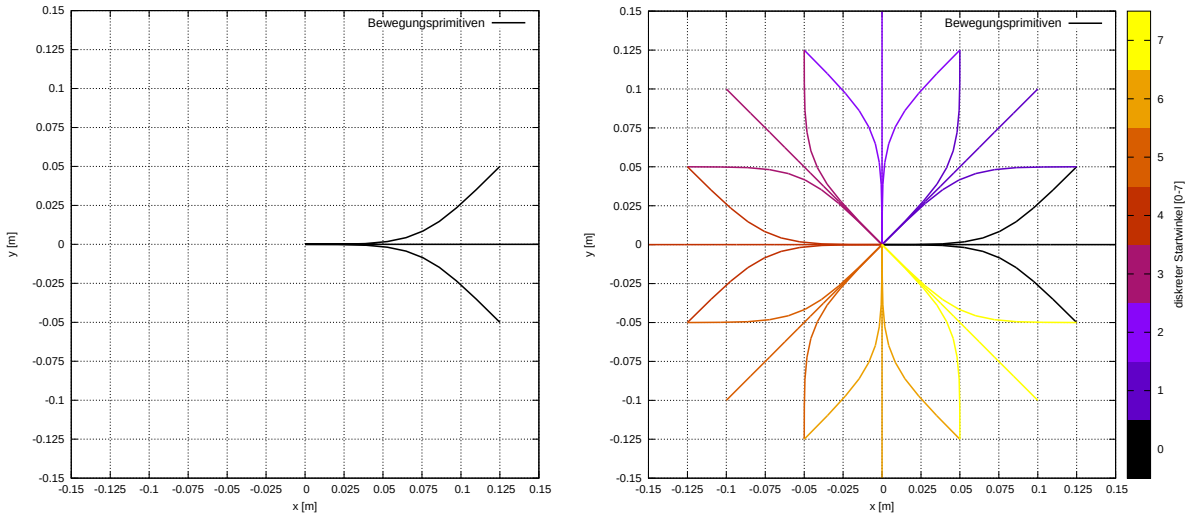



**Abbildung 5.7:** Winkeldiskretisierung der SBPL. Es werden 16 Winkelschritte genutzt, die jeweils einen $\delta = 22,5°$ großen Bereich repräsentieren. Der diskrete Winkel 0 deckt dabei den Bereich $[-11,25°; 11,25°)$ ab, mathematisch positiv gefolgt von den restlichen Bereichen.

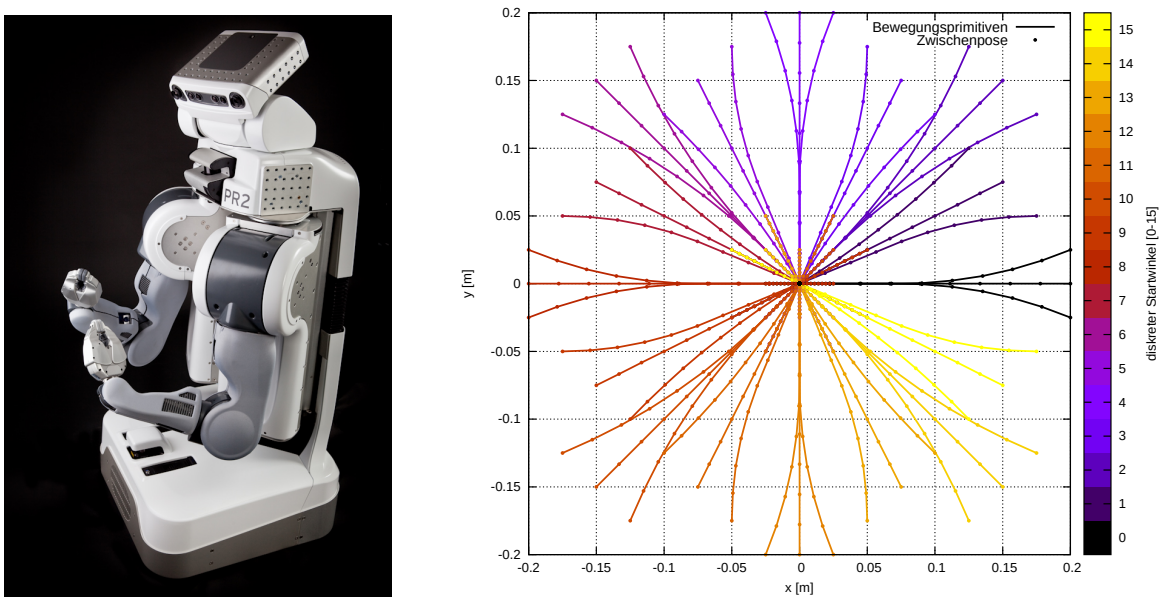
Zur Kollisionserkennung und -vermeidung werden für alle Bewegungsprimitiven Kollisionsmasken erstellt. Dafür wird der oft als Fußabdruck bezeichnete Perimeter des Roboters an Start-, End- und an jede Zwischenpose rotiert und translatiert und in ein Raster mit derselben Auflösung der 2D-Umgebungskarte diskretisiert. Diese Kollisionsmasken beschreiben für jede Bewegungsprimitive die Zellen, die bei der Ausführung der Bewegung durch einen mobilen Roboter berührt werden. Aus Performanzgründen werden die Masken einmalig bei der Initialisierung eines Planers berechnet und zwischengespeichert. Bei der Ausführung müssen diese für eine Kollisionserkennung nur an die entsprechenden Positionen translatiert werden. Die Translation der Masken erfordert im Vergleich zu ihrer Berechnung wenig Rechenaufwand.




(a) Bewegungsprimitiven für Startwinkel 0°.

(b) Kompletter Satz an Bewegungsprimitiven für 8 Startwinkel.

**Abbildung 5.8:** Beispiel für Bewegungsprimitiven. In diesem Beispiel mit acht diskreten Winkeln sind nur Vorwärtsbewegungen enthalten. Die einzelnen Primitiven starten und enden alle auf dem diskreten Raster. Die Primitiven für den diskreten Startwinkel 0 sind in (a), der volle Satz von Primitiven für 8 diskrete Winkel in (b) gezeigt.




(a) Der PR2 Roboter [104].

(b) Bewegungsprimitiven des PR2 aus der SBPL für alle 16 Startwinkel.

**Abbildung 5.9:** Bewegungsprimitiven mit Zwischenposen für den PR2 Roboter von *Willow Garage* aus der SBPL.

### Umgebungsrepräsentation

In der SBPL sind verschiedene Planungsumgebungen implementiert [57]:

- **2D** $(x, y)$ **rasterbasierte Planung** für die Planung eines translatierenden Roboters in einer zweidimensionalen, als Raster repräsentierten Ebene.
- **3D** $(x, y, \theta)$ **rasterbasierte Planung mit 2D Kollisionsvermeidung** $(x, y)$ für in einer Ebene translatierende und rotierende Roboter. Die Zustände werden in einem dreidimensionalen Raster modelliert. Als Hinderniskarte dient ein zweidimensionales Raster.
- **3D** $(x, y, \theta)$ **rasterbasierte Planung mit 3D Kollisionsvermeidung** $(x, y, z)$ für in einer Ebene translatierende und rotierende Roboter. Die Zustände werden in einem dreidimensionalen Raster modelliert. Als Hinderniskarte dient ein dreidimensionales Raster.
- **N-DOF ebene Planung** für einen Roboterarm mit $n$ Freiheitsgraden.

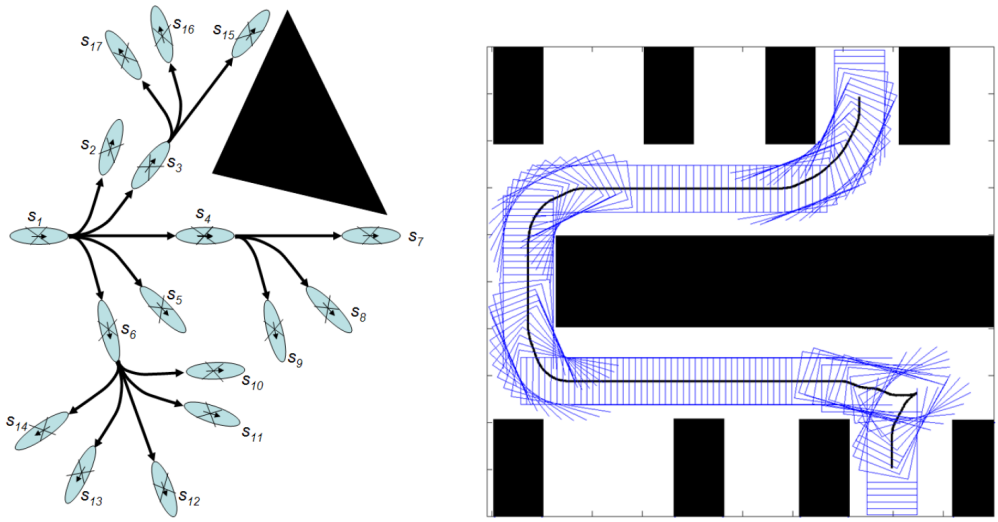
Für die Feldfahrzeuge im Anwendungsfall Infield-Transportlogistik wird die *3D rasterbasierte Planungsumgebung mit 2D Kollisionsvermeidung* eingesetzt. Die Hinderniskarte wird in einem zweidimensionalen Raster diskretisiert. Die einzelnen Zellen können Werte von 0 bis 255 einnehmen. Ein Parameter der Planungsumgebung legt fest, ab welchem Wert eine Zelle als Hindernis angesehen wird.

### Erzeugung des Graphen

In Abbildung 5.10 ist die Generierung eines Graphen in einem Beispiel dargestellt. Von einem Startzustand $S_1$ aus werden anhand von einer zweidimensionalen Hinderniskarte, Bewegungsprimitiven und deren Kollisionsmasken gültige Bewegungen identifiziert und deren Endzustände als Knoten in den Graphen aufgenommen. Die Zustände des Rasters werden also durch Knoten in einem gerichteten Graphen repräsentiert. Knoten aufeinander folgender Zustände werden durch Kanten verbunden. Die Kosten der erreichten Zustände ergeben sich aus Kosten des Vorgängers, Kosten der Bewegung und der Summe der Werte der durch die Kollisionsmaske abgedeckten Zellen in der Hinderniskarte. Die als Knoten neu aufgenommenen Zustände werden wiederum als Ausgang für weitere Bewegungen genutzt. In dem Beispiel aus Abbildung 5.10(a) sind dies die Zustände $S_2$ - $S_6$. Diese Expansion wird so lange fortgeführt, bis ein Zustand erreicht wird, der einer gesuchten Zielpose entspricht.

Das Ergebnis ist ein gerichteter Graph, in dem die Start- und die Zielpose als Knoten repräsentiert sind. Diese Parameter werden an einen Planer weitergegeben, der in dem Graphen den kürzesten Pfad sucht und aus der Knotenreihenfolge und den genutzten Bewegungsprimitiven einen gültigen Pfad erzeugt. In Abbildung 5.10(b) ist ein solcher Pfad exemplarisch dargestellt.

Die Bewegungsplanung wird bei diesem Verfahren nicht im Konfigurations- , sondern im Arbeitsraum durchgeführt. Die Planung im Arbeitsraum ist in der mobilen Robotik weit verbreitet, wie in Abschnitt 3.1.3 beschrieben und begründet.




(a) Beispiel eines Graphen, der durch Bewegungsprimitiven erzeugt wird [59].

(b) Ein vom Planer erzeugter, gültiger Pfad [59].

**Abbildung 5.10:** Beispiel des rasterbasierten Planers [59]. In (a) wird das Erzeugen eines Graphen anhand von 2D Hinderniskarte, Bewegungsprimitiven und Kollisionsmasken gezeigt. In diesem Graphen wird durch den Planer mit implementierten Suchverfahren ein kürzester, gültiger Pfad gesucht. Ein Beispiel für einen solchen Pfad ist in Abbildung (b) gezeigt.

Folgende Suchalgorithmen sind in der SBPL implementiert [57]:

- **ARA*** - anytime Variante von A*
- **Anytime D*** - anytime inkrementelle Variante von A*
- **R*** - randomisierte Version von A* (Hybrid aus deterministischer und abtastender Suche)

Der Suchalgorithmus Anytime D* ist als inkrementelle Version von A* insbesondere auf Planannpassungen bei Änderungen in dynamischen Umgebungen ausgelegt. Da in der Bewegungsplanung immer zur Laufzeit Teilausschnitte des Feldes als Rasterkarten generiert werden, die während der Teilplanung als statisch angenommen werden können, wird dieser Algorithmus nicht eingesetzt.

Der randomisierte Ansatz von $R^*$ kann zu unterschiedlichen Ergebnissen für dieselben Planungsinstanzen führen. Das ist in dem Infield-Transportlogistik Szenario nicht erwünscht, weil sich die Fahrzeuge in ähnlichen Situationen möglichst konsistent und vorausschaubar verhalten sollen. Ein solches Verhalten ist generell sinnvoll und erstrebenswert in Prozessen, in denen menschliche Anwender mit mobilen (teil-)autonomen Maschinen zusammenarbeiten.

Mit dem ARA* Algorithmus kann in kurzer Zeit ein initiales Planungsergebnis gefunden werden und durch die anytime Implementierung in einem definierten Zeitraum durch ein besseres Ergebnis ersetzt werden. Daher ist dieser Planer besonders geeignet und wird in dem entwickelten Planungssystem für die Bewegungsplanung der Feldfahrzeuge genutzt.

### 5.2.2 Approximation einer Referenztrajektorie mit Bewegungsprimitiven

Die Feldfahrzeuge sollen im Bereich der inneren Feldgrenze im Voraus berechnete Fahrspuren nutzen. Diese Fahrspuren werden geometrisch durch paralleles Verschieben eines Referenzlinienzuges erstellt (siehe Abschnitt 4.1.4), ohne die kinematischen Einschränkungen der beteiligten Maschinen zu berücksichtigen. Deshalb ist nicht gewährleistet, dass die Fahrspuren in geforderter Genauigkeit durch die Maschinen abgefahren werden können. Die Fahrspuren, oder Teile davon, können also nicht direkt als Sollstrecke für die Fahrzeuge genutzt werden, sondern müssen vorher noch validiert und gegebenenfalls angepasst werden.

In der Literatur finden sich Algorithmen, die durch Pfadglättung für kontinuierliche Krümmungsverläufe und die Abfahrbarkeit von Sollstrecken sorgen (siehe Abschnitt 3.1.3). In dem vorliegenden Anwendungsfall sollten alle Verfahren zur Bewegungsplanung jedoch dieselbe kinematische Beschreibung der Fahrzeuge nutzen. Für die Bewegungsprimitiven der SBPL gab es bislang kein Verfahren zum Erstellen von abfahrbaren Trajektorien für vorgegebene Sollstrecken. Daher wurde im Rahmen dieser Arbeit ein neues Verfahren entwickelt [86], das eine Sollstrecke mit Bewegungsprimitiven approximiert, ohne dabei einen Toleranzbereich um die Sollstrecke zu verlassen. In Abbildung 5.11 ist der Ansatz des Verfahrens in einem Blockdiagramm dargestellt.

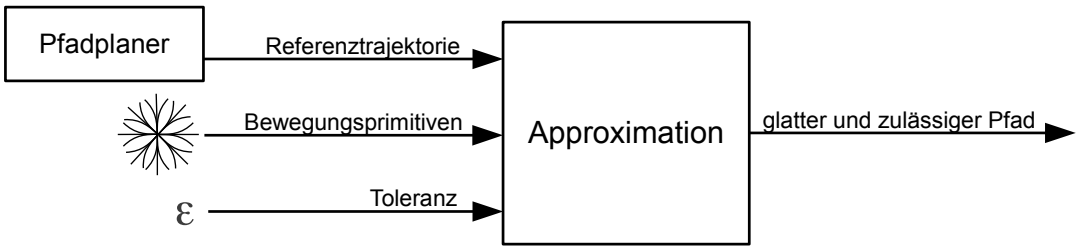



**Abbildung 5.11:** Blockdiagramm des Approximationsalgorithmus [86]. Eine Referenztrajektorie wird von einem Planer ohne Berücksichtigung der kinematischen und dynamischen Einschränkungen des Fahrzeugs erstellt. Die Trajektorie wird innerhalb eines Toleranzbereiches mit Abstand $\epsilon$ um die Trajektorie mit Hilfe von Bewegungsprimitiven approximiert. Da diese Bewegungsprimitiven ausschließlich gültige Bewegungen repräsentieren und ohne Orientierungssprünge verbunden werden, ist der entstehende Pfad glatt und gültig.

Ziel der Approximation einer Referenztrajektorie ist es, einen ähnlichen, glatten und durch die Fahrzeuge abfahrbaren Pfad zu generieren, der einen Toleranzbereich um die Trajektorie nicht verlässt. Dafür wird die Startpose in dem Wurzelknoten eines Baumes gespeichert. Der Baum wird mit Hilfe der Bewegungsprimitiven in A* Manier entlang der Referenztrajektorie expandiert. Es wird vorausgesetzt, dass der vorgeschaltete Pfadplaner sicherstellt, dass ein Pfad, wenn er in dem Toleranzbereich um die Referenztrajektorie liegt, kollisionsfrei ist. Daher muss bei der Erstellung des Graphen keine kostenintensive Kollisionserkennung vorgenommen werden.

Für die Approximation werden Bewegungsprimitiven in dem SBPL Format genutzt. Die Menge der Bewegungsprimitiven, die zum Expandieren eines Knotens genutzt werden können, ist beschränkt auf diejenigen, deren Startwinkel der Orientierung der durch den Knoten repräsentierten Pose entspricht.

**Algorithmus 5** Approximation einer Referenztrajektorie $w$ mit Bewegungsprimitiven $M$.

**Prozedur** APPROXIMATION($w$, $M$)
  initialisiere leeren Baum $G$
  initialisiere leere aktive Knoten Liste $L$
  initialisiere leere Zielknoten Liste $Z$
  füge Wurzelknoten $v_{root}$ zu $G$ und zu $L$ hinzu
  **solange** $L$ hat Elemente **und** Zeitlimit nicht überschritten **führe aus**
    wähle besten Knoten $v_{best}$ aus $L$ nach $f(v) = \alpha \cdot h(v) + g(v)$
    expandiere besten Knoten $v_{best}$ mit $m \in M$, wobei $\theta_{disc}(m_{start}) = \theta_{disc}(v_{best})$
    passe $\alpha$ für die nächste Iteration an
  **Ende solange**
  wähle besten Zielknoten $v_{best\ target}$ aus der Zielknotenliste $Z$
  rekonstruiere Pfad aus $v_{best\ target}$ durch Nutzung des Vorgänger Attributs $pred(v)$
**Ende Prozedur**

In Algorithmus 5 ist das Verfahren der Approximation einer Referenztrajektorie mit Bewegungsprimitiven aufgelistet. Es wird ein leerer Baum, sowie eine leere aktive Knoten- und Zielknotenliste initialisiert. Der Wurzelknoten, der die Startpose des Roboters repräsentiert, wird in den Graphen und in die aktive Knotenliste aufgenommen. Ausgehend von diesem Knoten wird der Baum expandiert. Dazu wird in jeder Iteration der beste Knoten der aktiven Knotenliste als zu expandierender Knoten ausgewählt. Bei der Auswahl werden bisherige Kosten

$$g(v) = g(pred(v)) + d_{vs_n} + c_{Bewegung} + k|\delta| \tag{5.1}$$

bis zu einem Knoten und geschätzte Kosten

$$h(v) = d_{vs_{n+1}} + \sum_{i=n+1}^{|w|-1} len(s_i) \tag{5.2}$$

von einem Knoten bis zum Ziel berücksichtigt. Die Kostenfunktion ist in Abbildung 5.12 und die Heuristikfunktion in Abbildung 5.13 dargestellt.

Abbildung 5.14 zeigt ein Beispiel des beschriebenen Approximationsverfahrens. Die Expansion des Wurzelknotens, der die Startpose repräsentiert, ist in Abbildung 5.14(b) dargestellt. Dazu werden alle Bewegungsprimitiven, die denselben diskreten Startwinkel haben wie die durch den Wurzelknoten repräsentierte Pose, an die diskrete Startposition verschoben. Die Bewegungsprimitiven, die den Toleranzbereich nicht verlassen (in Abbildung 5.14(b) grün dargestellt), sind gültig und deren Endposen werden als neue Knoten in den Graphen und in die aktive Knotenliste aufgenommen. Dabei werden die Kosten $g(v)$ (Formel (5.1)) der neuen Knoten vom Wurzelknoten an und die geschätzten Kosten $h(v)$ (Formel (5.2)) entlang der Referenztrajektorie bis zur Zielpose berechnet und als Attribute zusammen mit dem Vorgängerknoten gespeichert. Bewegungsprimitiven, die zum Verlassen des Toleranzbereiches führen, sind ungültig und werden verworfen. Aus den nach der Iteration im Graphen vorhandenen Knoten wird nach der Formel

$$f(v) = \alpha \cdot h(v) + g(v) \tag{5.3}$$

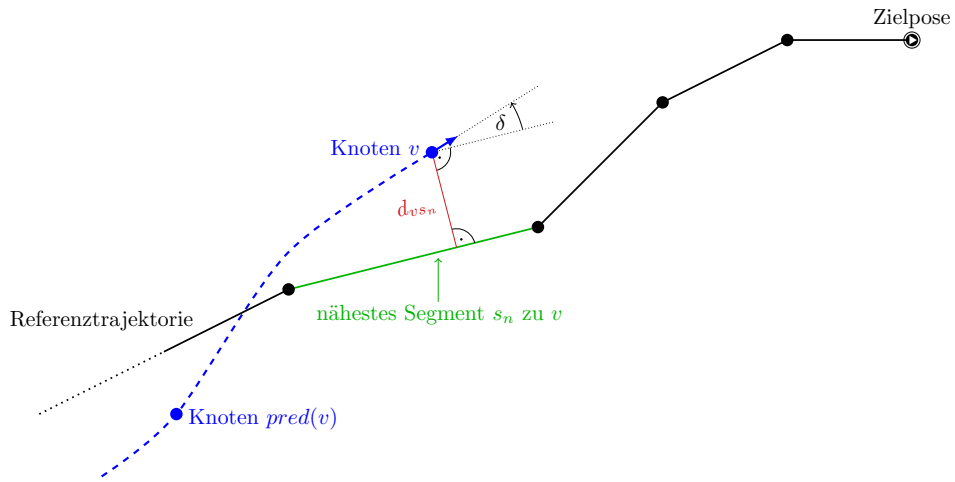



**Abbildung 5.12:** Kostenfunktion $g(v)$ der Approximation. Die Kosten eines Knotens $v$ berechnen sich aus den Kosten des Vorgängers $g(pred(v))$, der Distanz $d_{vs_n}$ zum nächsten Segment $s_n$, der Kosten für die Bewegung $c_{Bewegung}$ und dem Betrag der Orientierungsdifferenz $\delta$, wie in Formel (5.1) angegeben.

der beste Knoten ausgewählt. Der Parameter $\alpha$ ist dabei ein gewichteter Faktor, der zwischen Breiten- und Tiefensuche verlagern kann. Dieser Faktor wird in jeder Iteration angepasst. Der beste Knoten wird aus der aktiven Knotenliste gelöscht und analog zum Wurzelknoten expandiert (Abbildung 5.14(c)). Dieser Vorgang wird so lange wiederholt, bis die aktive Knotenliste leer ist oder ein Zeitlimit erreicht wurde. Wird dabei ein Knoten hinzugefügt, der in einem parametrierbaren Bereich um die Zielpose liegt, wird dieser der Zielknotenliste hinzugefügt. Nach dem Iterationsprozess wird aus der Zielknotenliste der beste Zielknoten ausgewählt. Von diesem Zielknoten wird der beste Pfad rückwärts mit Hilfe des Vorgängerattributs bestimmt und zurückgegeben (Abbildung 5.14(d)).

Mit dem beschriebenen Verfahren kann ein abfahrbarer Pfad generiert werden, der die Referenztrajektorie annähert. Ob eine Lösung für eine Probleminstanz gefunden werden kann, hängt von den Bewegungsprimitiven, der Form der Referenztrajektorie und der Größe des Toleranzbereiches ab.

Ein Schwachpunkt der beschriebenen Approximation ist, dass gerade Teilstücke der Referenztrajektorie kurvig approximiert werden, falls die Orientierung der geraden Teilstücke nicht einer diskreten Orientierung der Bewegungsprimitiven entspricht (siehe Abbildung 5.15). Auch eine drastische Erhöhung der Winkelauflösung würde das Problem nicht vollständig lösen, dabei aber den Suchraum stark erweitern und somit Laufzeit und Speicherbedarf negativ beeinflussen.

Um diesem ungewollten Verhalten entgegenzuwirken, kann vor der Approximation die Referenztrajektorie validiert werden. In diesem Validierungsschritt wird die Winkeldifferenz $\delta$ zwischen aufeinander folgenden Segmenten $s_a$ und $s_b$ in Abhängigkeit der Segmentlänge folgendermaßen bewertet (vergleiche Abbildung 5.16):

$$\xi = \frac{|\delta|}{l(s_a) + l(s_b)} \tag{5.4}$$

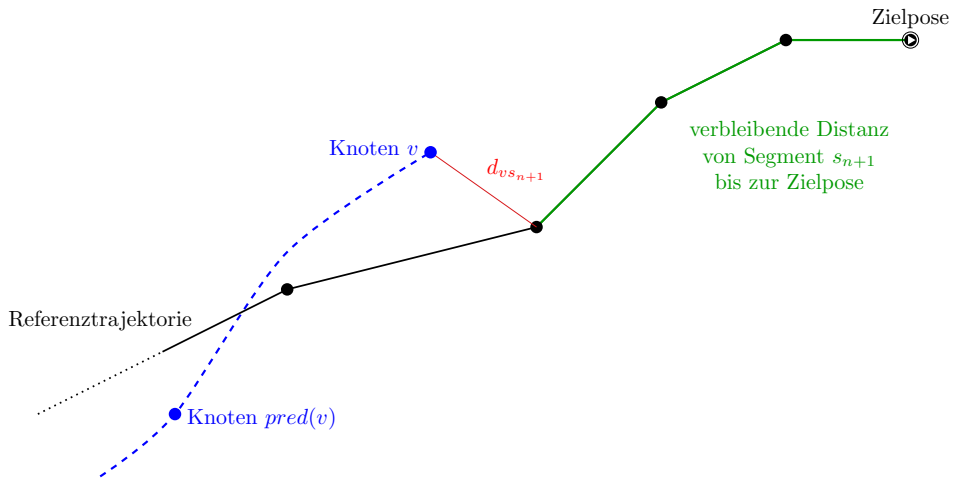
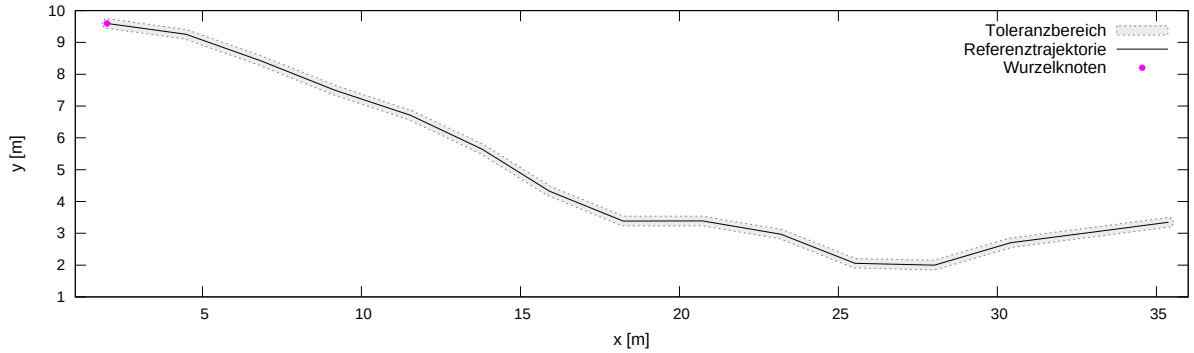
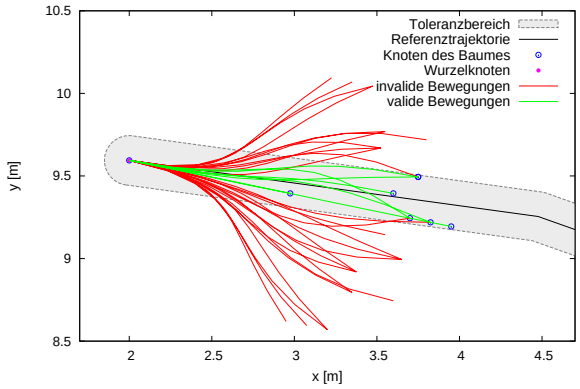



**Abbildung 5.13:** Heuristik $h(v)$ der Approximation. Die geschätzten Kosten bis zur Zielpose berechnen sich aus der Distanz $d_{vs_{n+1}}$ der Position des Knotens $v$ zum zweiten Punkt des am nächsten liegenden Segments und der Summe der Längen der bis zur Zielpose verbleibenden Segmente wie in, Formel (5.2) definiert.

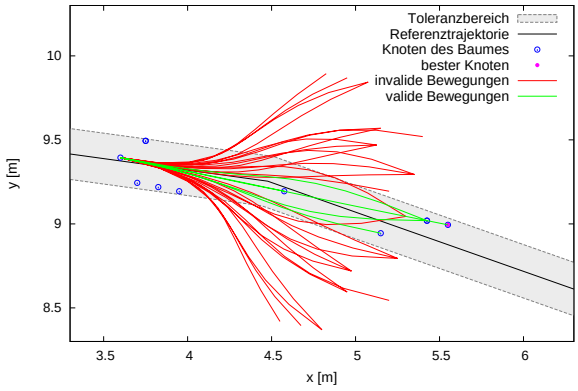
Wenn $\xi$ einen Schwellwert überschreitet, so sind diese Segmente ungültig und müssen approximiert werden. Segmente, für die $\xi$ diesen Schwellwert nicht unterschreitet, sind gültig und werden direkt als Wegpunkte übernommen. Dieser Validierungsprozess ist in Abbildung 5.17 gezeigt.



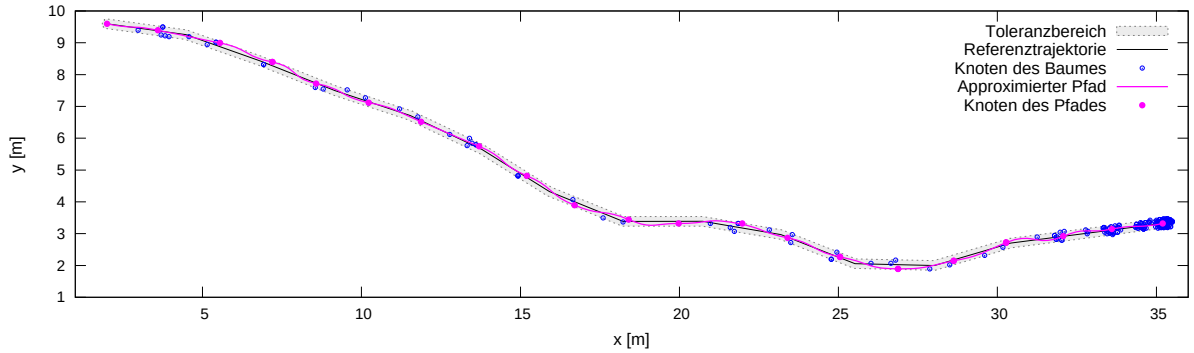

(a) Referenztrajektorie, Toleranzbereich und Wurzelknoten des Graphen.




(b) Expansion des Wurzelknotens. Endposen gültiger Bewegungen (grün) werden als Knoten (blau) in den Graphen aufgenommen, ungültige Bewegungen (rot) werden verworfen.




(c) Expansion des ausgewählten besten Knotens im zweiten Iterationsschritt.




(d) Knoten des Graphen nach Expansionsschleife (blau) und der rückwärts vom Zielknoten aus generierte beste Pfad (magenta).

**Abbildung 5.14:** Beispiel der Approximation einer Referenztrajektorie mit Bewegungsprimitiven innerhalb eines Toleranzbereiches (a). In (b) und (c) ist das Erstellen des Graphen dargestellt. In (d) sind die Knoten des erstellten Graphen, der beste Zielknoten und der approximierte Pfad gezeigt.

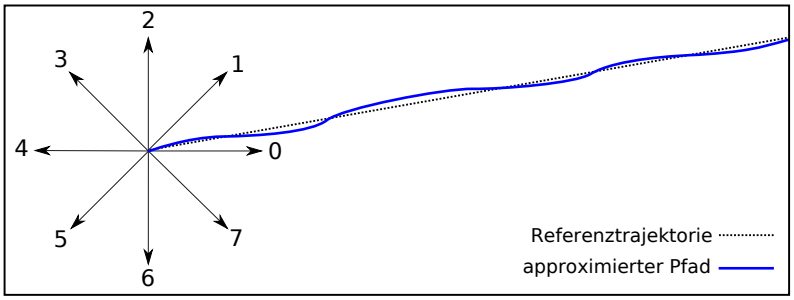



**Abbildung 5.15:** Einschränkungen durch Winkeldiskretisierung. Gerade Teile der Referenztrajektorie werden kurvig approximiert, wenn deren Orientierungen nicht genau einem diskreten Winkel entsprechen. In diesem Beispiel sind vereinfachend acht diskrete Winkel als Pfeile gezeigt.

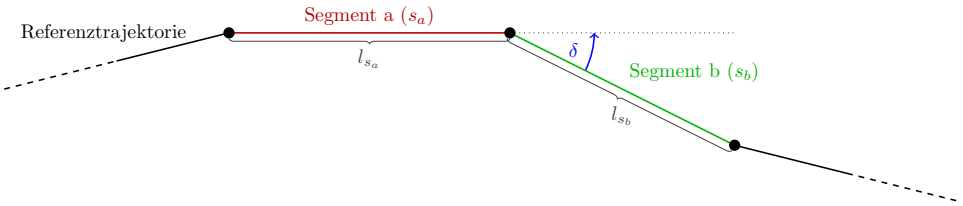



**Abbildung 5.16:** Validierung von aufeinander folgenden Segmenten anhand von Winkeldifferenz $\delta$ und Segmentlängen $l_{s_a}$ und $l_{s_b}$.

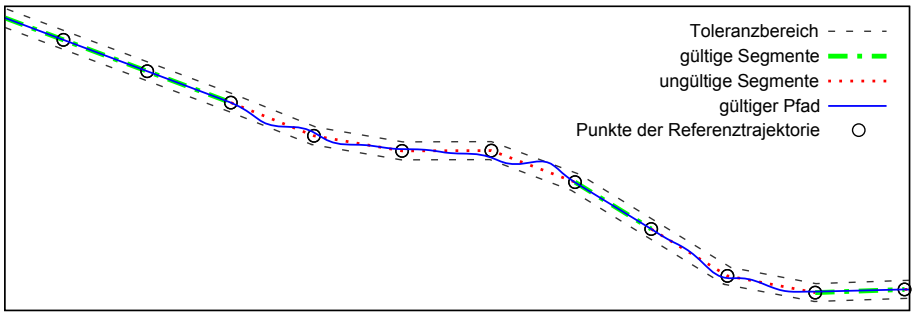



**Abbildung 5.17:** Validierung einer Referenztrajektorie. Wenn die Winkeldifferenz zwischen verbundenen Segmenten kleiner als ein Schwellwert ist, sind die Segmente gültig und werden als Wegpunkte genutzt, andernfalls werden die Segmente durch den beschriebenen Algorithmus approximiert.

## 5.3 Bewegungsplanung für den Mähdrescher

In dem Projekt *marion* wurde ein Mähdrescher des Typs *CLAAS LEXION 760* eingesetzt (siehe Abschnitt 2.2). Die grundlegende Kinematik des Fahrzeugtyps wird in Abschnitt 5.3.1 kurz erläutert. In Abschnitt 5.3.2 wird die Bestimmung der Bewegungsprimitiven für den Mähdrescher beschrieben.

Durch das in Abschnitt 4.1.2 beschriebene Freischneiden des Vorgewendes zu Beginn eines Ernteprozesses entsteht Platz für Wendemanöver der Feldfahrzeuge. Die verbleibende, nicht abgeerntete Fläche wird beschrieben durch die umschließende innere Feldgrenze. Dieser Bereich wird in Fahrspuren unterteilt (siehe Abschnitt 4.1.4). Innerhalb der inneren Feldgrenze soll der Mähdrescher die vorgegebenen Fahrspuren nutzen, im Vorgewende darf er sich auf der abgeernteten Fläche frei bewegen. Dieses zonenbasierte Verhalten führt zu zwei unterschiedlichen Planungsaufgaben. In Abschnitt 5.3.3 wird die Bewegungsplanung auf einer Fahrspur und in Abschnitt 5.3.4 die Bewegungsplanung im Vorgewende beschrieben.

### 5.3.1 Kinematik des Mähdreschers

Das kinematische Modell des Mähdreschers ist in Abbildung 5.18 dargestellt. Die hintere Achse wird als Lenkachse verwendet, während die vordere Achse starr ist. Das kinematische Modell entspricht somit der Ackermann Kinematik in der Rückwärtsfahrt. Bedingt durch die starre Frontachse kann der Mähdrescher nicht auf der Stelle drehen. Deshalb ist ein kontinuierlicher Lenkwinkelverlauf eine wichtige Anforderung an einen abfahrbaren Pfad. Aus dem Einschlagwinkel eines Rades, beispielsweise $\delta_2$ und dem Achsabstand $a$ berechnet sich der momentan gefahrene Kurvenradius als

$$R = \frac{a}{tan\,\delta_2} \tag{5.5}$$

Für den Einschlagwinkel des zweiten Rades gilt mit der Spurbreite $b$

$$tan\,\delta_1 = \frac{a}{R+b}, \tag{5.6}$$

mit (5.5)

$$tan\,\delta_1 = \frac{a}{\frac{a}{tan\,\delta_2}+b}. \tag{5.7}$$

Da die Interna des Lenksystems nicht bekannt sind (siehe Abschnitt 3.2.1), müssen in Experimenten dessen Performanz und Reaktion auf Sollstrecken analysiert werden. Anhand der Analyseergebnisse werden die wesentlichen kinematischen Eigenschaften abgebildet und für die Bewegungsplanung in Form von Bewegungsprimitiven gespeichert (Abschnitt 5.3.2). So soll sichergestellt werden, dass das Lenksystem die von der Bewegungsplanung generierten Pfade in geforderter Genauigkeit abfahren kann.

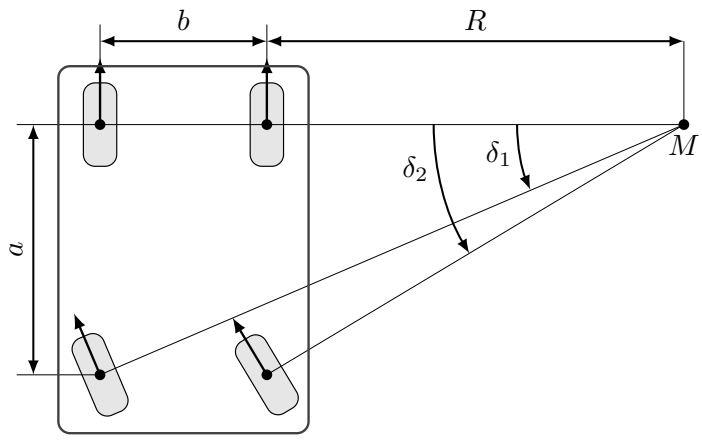



**Abbildung 5.18:** Kinematik des Mähdreschers. Die hintere Achse wird als Lenkachse verwendet. Das kinematische Modell entspricht der Ackermann Kinematik in der Rückwärtsfahrt.

### 5.3.2 Bestimmung von Bewegungsprimitiven

Auf Basis der experimentell ermittelten Parameter des Lenksystems (siehe Abschnitt 6.1) werden für die beiden eingesetzten Verfahren zur Bewegungsplanung aus Abschnitt 5.2 Bewegungsprimitiven erzeugt. Es werden unterschiedliche Sätze generiert, weil für die Prozesszustände Dreschen auf einer Fahrspur und Wenden im Vorgewende

1. unterschiedliche Geschwindigkeiten angenommen werden müssen,
2. unterschiedliche Fahrzeugverhalten erwünscht sind und daher
3. unterschiedlich ausgeprägte Suchräume exploriert werden sollen.

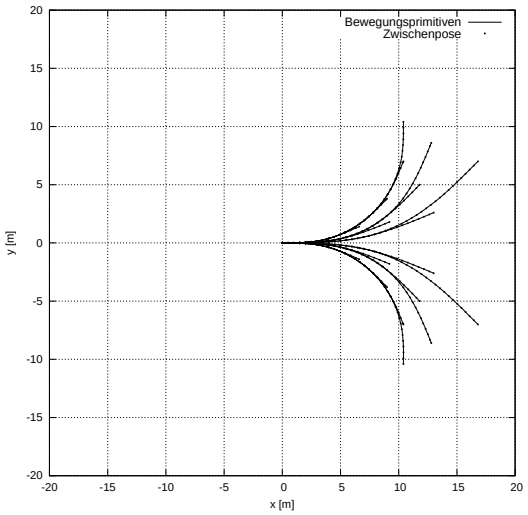
Durch unterschiedliche Geschwindigkeiten und die vorgegebene maximale Lenkwinkeländerung pro Zeiteinheit entsteht eine unterschiedliche maximale Krümmungsänderung der Bewegungsprimitiven. Zusätzlich kann das Fahrzeugverhalten durch Kostenmultiplikatoren für einzelne Bewegungsprimitiven gesteuert werden. So können zum Beispiel unterschiedliche Wenderadien mit unterschiedlichen Kosten verbunden werden, so dass ein kürzerer Weg mit höherer Krümmung teurer wird als ein etwas längerer Pfad mit geringerer Krümmung. Das wäre beispielsweise sinnvoll, wenn dieser längere Pfad aufgrund kleinerer Lenkbewegungen mit einer höheren Geschwindigkeit abgefahren werden könnte und schneller zum Ziel führen würde. Beim Abfahren der Fahrspur sollen im Gegensatz dazu möglichst keine großen Lenkbewegungen durchgeführt werden. Daher werden für Fahrspuren Bewegungsprimitiven eingesetzt, die wesentlich weiter geradeaus gerichtet sind als die für Wendevorgänge genutzten.

Die beiden Sätze von Bewegungsprimitiven für die eingesetzten Verfahren werden direkt aus den ermittelten Lenkparametern per Python Skript generiert. Dabei werden die in Abschnitt 5.2.1 beschriebenen vorausgesetzten Eigenschaften für Bewegungsprimitiven eingehalten. Die wichtigsten Eigenschaften sind der Krümmungswert 0 zum Beginn und zum Ende jeder Primitiven, und dass jede Primitive an der diskreten Position (0,0) startet und an einer diskreten Position

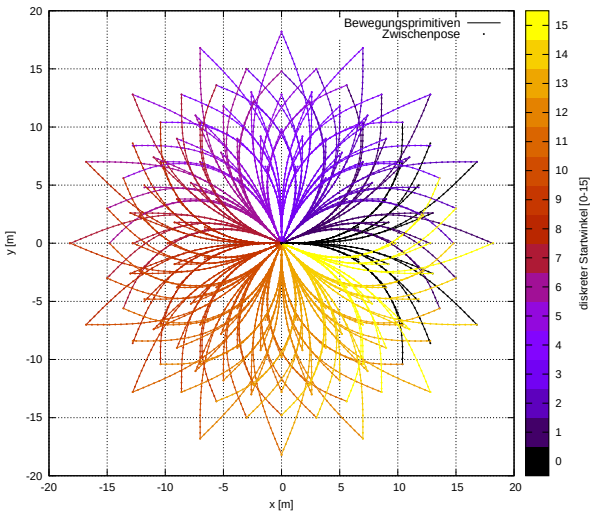
($x$·Zellenauflösung, $y$·Zellenauflösung), mit $x, y \in \mathbb{N}$, endet. Für die eingesetzten Fahrzeugtypen sind außerdem kontinuierliche Lenkwinkeländerungen innerhalb jeder Bewegungsprimitive nötig, um einen abfahrbaren Pfad planen zu können. In dem Skript sind die Anzahl an diskreten Winkelschritten und die zu erreichenden diskreten Orientierungen aller Endposen für den diskreten Startwinkel 0 vorgegeben. Die diskreten Endpositionen können aus den Lenkparametern ermittelt werden. Die Primitiven werden dann für alle Winkelschritte rotiert und so angepasst, dass alle Endposen auf dem diskreten Gitter liegen. Die Bewegungsprimitiven werden dabei aus geraden Segmenten, Klothoiden- und Kreisabschnitten generiert, so dass alle beschriebenen Eigenschaften eingehalten werden.

In Abbildung 5.19 sind die erstellten Bewegungsprimitiven für einen Mähdrescher des Typs CLAAS Lexion 760 für das Vorgewende (Abbildung 5.19(a) und Abbildung 5.19(b)) und für die Fahrspur (Abbildung 5.19(c) und Abbildung 5.19(d)) gezeigt. Abbildung 5.19(a) und Abbildung 5.19(c) zeigen dabei jeweils die Bewegungsprimitiven für den diskreten Startwinkel 0. Hier wird das erwünschte unterschiedliche Fahrzeugverhalten deutlich. Während bei der Planung im Vorgewende Wendemanöver im Vordergrund stehen, sind für die Planung auf einer Fahrspur wesentlich weniger ausholende Fahrmanöver hinterlegt. Außerdem wird im Vergleich von Abbildung 5.19(b) und Abbildung 5.19(d) deutlich, dass für die Planung im Vorgewende 16 diskrete Winkelschritte berücksichtigt werden, während bei der Planung auf einer Fahrspur die doppelte Anzahl an Winkelschritten gewählt wurde. So kann beim Ernten auf der Fahrspur eine wesentlich höhere Genauigkeit in der Planung und Planausführung erreicht werden, als bei Wendemanövern im Vorgewende. Das Erreichen einer hohen Genauigkeit auf einer Fahrspur ist erforderlich, um die Anforderung einer flächendeckenden Abarbeitung (siehe Abschnitt 2.2) erfüllen zu können. Hohe Abweichungen können dazu führen, dass Streifen der inneren Feldfläche nicht abgeerntet werden.

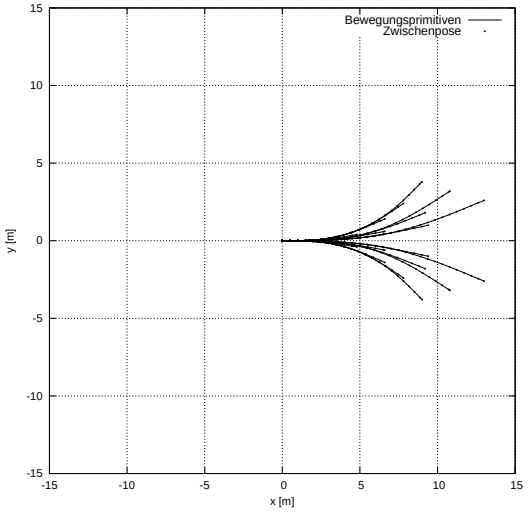
Beide Sätze an Bewegungprimitiven aus Abbildung 5.19 enthalten ausschließlich Vorwärtsbewegungen. Eine Rückwärtsfahrt ist in dem betrachteten Anwendungsfall der Infield-Transportlogistik unerwünscht, da dies zum Beenden der Lenkautomatik führen würde. Algorithmisch wäre das Einplanen von Rückwärtsfahrten durch das Hinzufügen von rückwärts gerichteten Bewegungsprimitiven leicht integrierbar.



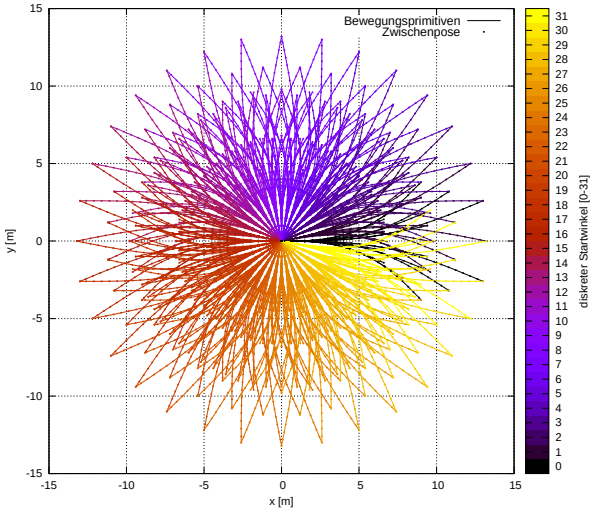

(a) Auszug der Bewegungprimitiven des Mähdreschers für den SBPL Planer und den diskreten Startwinkel 0.




(b) Vollständiger Satz an Bewegungprimitiven des Mähdreschers für den SBPL Planer mit allen 16 diskreten Startwinkeln.




(c) Auszug der Bewegungprimitiven des Mähdreschers für die Approximation von Referenztrajektorien und den diskreten Startwinkel 0.




(d) Vollständiger Satz an Bewegungprimitiven des Mähdreschers für die Approximation von Referenztrajektorien mit allen 16 diskreten Startwinkeln.

**Abbildung 5.19:** Die Bewegungsprimitiven für den Mähdrescher. Die zur Bewegungsplanung eingesetzten Verfahren aus Abschnitt 5.2 benötigen unterschiedlich ausgeprägte Bewegungprimitiven. In (a) und (b) sind die Bewegungsprimitiven für den im Vorgewende genutzten SBPL Planer (Abschnitt 5.2.1), in (c) und (d) die Bewegungprimitiven für das auf Fahrspuren genutzte Approximationsverfahren (Abschnitt 5.2.2) gegeben. (a) und (c) zeigen jeweils die Bewegungsprimitiven für den diskreten Startwinkel 0, während (b) und (d) den vollen Satz der Bewegungsprimitiven für alle 16 beziehungsweise 32 diskreten Startwinkel zeigt.

### 5.3.3 Bewegungsplanung auf einer Fahrspur

Innerhalb der inneren Feldfläche soll der Mähdrescher die Fahrspuren nutzen. Diese Fahrspuren decken die innere Feldfläche für die angegebene Arbeitsbreite des Mähdreschers komplett ab. Die benachbarten Fahrspuren überlappen sich dabei leicht, um bei kleinen Abweichungen von der Fahrspur keine Fläche auszulassen. Da die Fahrspuren geometrisch erzeugt werden, ohne die kinematischen Eigenschaften der Fahrzeuge zu berücksichtigen (Abschnitt 4.1.4), müssen die Fahrspuren aber noch in abfahrbare Sollstrecken transformiert werden. Die Fahrspuren sollen dabei aber durch die Sollstrecken so genau wie möglich repräsentiert werden. Daher kommt für die Bewegungsplanung des Mähdreschers auf einer Fahrspur die in Abschnitt 5.2.2 beschriebene Approximation einer Referenztrajektorie mit Bewegungsprimitiven zum Einsatz. Dabei wird je nach der Lage des nächsten Routenpunkts ein Ausschnitt einer Fahrspur oder eine gesamte Fahrspur als Referenztrajektorie an den Planer übergeben. Außerdem werden die Bewegungsprimitiven aus Abschnitt 5.3.2 und die halbe Überlappung der Fahrspuren als Toleranzgröße übergeben. Bei der Approximation wird keine Hindernisvermeidung durchgeführt, da davon ausgegangen wird, dass keine Kollisionen mit Hindernissen auftreten können, solange ein Toleranzbereich einer definierten Größe nicht überschritten wird. Eine Referenztrajektorie wird in dem ersten Schritt auf nicht abfahrbare Teilstrecken untersucht (siehe Abschnitt 5.2.2). Werden keine solchen Teile gefunden, so werden die allesamt gültigen Punkte der Referenztrajektorie als Ergebnis der Bewegungsplanung genutzt. Andernfalls werden die ungültigen Teiltrajektorien einzeln approximiert und anschließend gemeinsam mit den gültigen Segmenten als Ergebnis der Bewegungsplanung zurückgegeben (siehe Abbildung 5.17).

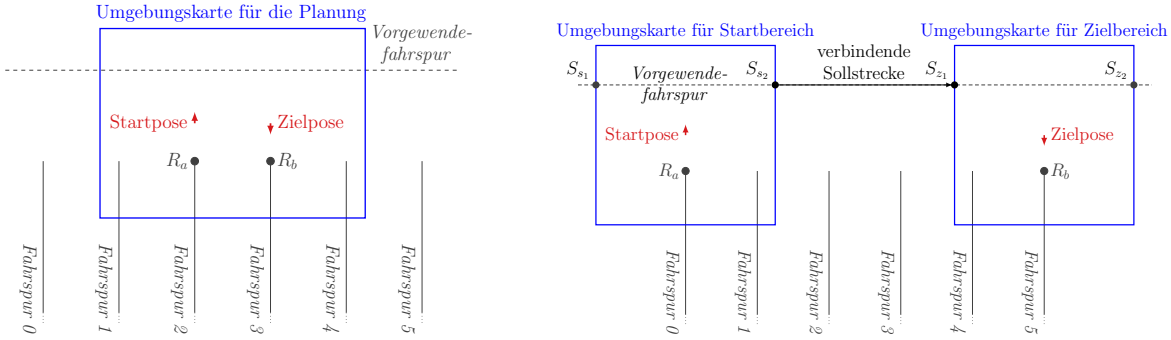
### 5.3.4 Bewegungsplanung im Vorgewende

Das Vorgewende ist der zu Beginn des Ernteprozesses freigeschnittene Bereich zwischen der äußeren und der inneren Feldgrenze (siehe Abschnitt 2.2). Dieser Bereich wird von dem Mähdrescher für Wendemanöver bei dem Wechsel zwischen Fahrspuren genutzt. Für die Bewegungsplanung des Mähdreschers wird der Bereich außerhalb der äußeren Feldgrenze und der Bereich innerhalb der inneren Feldgrenze als Hindernis betrachtet. Für die Planung der Wendemanöver soll der gitterbasierte Planer der SBPL (siehe Abschnitt 5.2.1) eingesetzt werden. Der Planer benötigt neben den Bewegungsprimitiven eine Rasterkarte der Umgebung. Die Auflösungen der Rasterkarte und der Bewegungsprimitiven müssen identisch sein. Wie in Abschnitt 3.2 beschrieben, sollen die Fahrspuren möglichst genau abgefahren werden. Daher ist wichtig, dass die Einfahrt in eine Fahrspur, die durch die Planung des Wendemanövers vorgegeben wird, möglichst genau auf die Fahrspur führt. Die Distanz zur Fahrspur und der Orientierungsfehler sollten also so gering wie möglich sein. Daher ist eine hohe Zellauflösung für Bewegungsprimitiven und Rasterkarte zu wählen. Eine Planung in einer Rasterkarte, die das gesamte Feld repräsentiert, ist aufgrund dieser hohen Auflösung und der Ausmaße von Feldern sehr rechen- und speicherintensiv. Daher wird zur Laufzeit nur ein relevanter Ausschnitt für ein Wendemanöver betrachtet, der Start- und Zielpose beinhaltet. Da je nach Feldgeometrie auch ein einzelner Wendevorgang eine große Umgebungskarte erfordern kann, wird eine weitere Unterteilung in einen Start- und Zielbereich unternommen. Für die Verknüpfung der Bereiche wird die Vorgewendefahrspur genutzt. Für

längere Fahrten im Vorgewende wird so immer dieselbe Fläche überfahren, was aus Sicht des Prozesses erwünscht ist.

### Bestimmung der Planungsumgebung

Zur Bestimmung der Planungsumgebung, in der von dem Planer ein Pfad für den Fahrspurwechsel des Mähdreschers gesucht werden soll, wird die Start- und die Zielpose benötigt. Bei Wendemanövern des Mähdreschers sind diese durch zwei Routenpunkte vorgegeben, die zu unterschiedlichen Fahrspuren gehören. In dem Beispiel aus Abbildung 5.20(a) sind dies die Routenpunkte $R_a$ und $R_b$. Dabei beschreibt $R_a$ die Ausfahrt aus Fahrspur 2 und $R_b$ die Einfahrt in Fahrspur 3. Da das Schneidwerk des Mähdreschers bei der Einfahrt in eine Fahrspur abgesenkt und bei der Ausfahrt angehoben werden muss, wird die Fahrspur um sechs Meter in das Vorgewende verlängert (siehe Abbildung 5.20). So bleibt dem Fahrer für das Verstellen des Schneidwerks ausreichend Zeit. Um die so erhaltene Start- und Zielpose, für die ein Pfad gesucht werden soll, wird das minimale umschreibende achsenparallele Rechteck mit einem Abstand von $d = 25$ m gelegt. Überschreiten Höhe und Breite des Rechtecks den Schwellwert $d_{max} = 85$ m nicht, so wird keine weitere Unterteilung vorgenommen. Dieser Fall ist in Abbildung 5.20(a) dargestellt. Das erhaltene Rechteck beschreibt in diesem Fall die Ausmaße der Umgebung, in der mit dem gitterbasierten Planer der SBPL geplant wird. Wird der Schwellwert jedoch überschritten, so bilden achsenparallele Rechtecke mit dem Abstand $d$ in x- und in y-Richtung um beide Posen zwei getrennte Planungsumgebungen für den Start- und den Zielbereich des Fahrspurwechsels. Diese Bereiche müssen die Vorgewendefahrspur wie in Abbildung 5.20(b) dargestellt schneiden. Die kürzeste Verbindung zwischen zwei Schnittpunkten der Bereiche auf der Vorgewendefahrspur wird ermittelt und als Sollstrecke zur Verbindung der Bereiche genutzt. Im Startbereich muss zu dem Start der Sollstrecke und im Zielbereich von dem Ende der Sollstrecke aus geplant werden. In dem Start- und Zielbereich kommt der gitterbasierte Planer zum Einsatz, während für die Sollstrecke das in Abschnitt 5.2.2 beschriebene Verfahren genutzt wird. Die Ergebnisse der einzelnen Planungsschritte werden zu einem Gesamtpfad konkateniert und als Ergebnis der Bewegungsplanung im Vorgewende zurückgegeben.




(a) Start- und Zielpose liegen in derselben Planungsumgebung.

(b) Start- und Zielpose liegen in unterschiedlichen Planungsumgebungen, die durch eine Sollstrecke verbunden sind.

**Abbildung 5.20:** Bestimmung der Planungsumgebung für Wendemanöver des Mähdreschers. Um Start- und Zielpose wird ein umschreibendes achsenparalleles Rechteck mit einem definierten Rahmen gelegt. Unterschreiten Länge und Höhe des Rechtecks einen Schwellwert, so beschreibt das Rechteck die Planungsumgebung (a). Wird der Schwellwert überschritten, so folgt eine Unterteilung in einen Start- und einen Zielbereich (b). Ein Abschnitt der Vorgewendefahrspur dient als Sollstrecke zur Verbindung der beiden Bereiche.

### Berechnung der Umgebungskarte

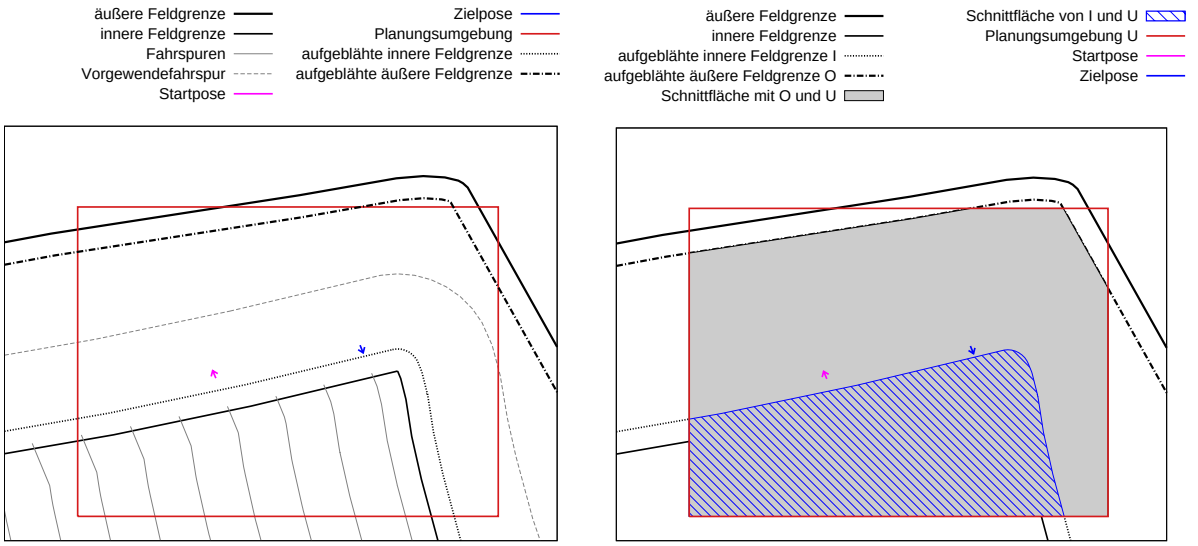
Für den gitterbasierten Planer muss für jede Planungsinstanz eine Rasterkarte der Umgebung erstellt werden. In der Rasterkarte müssen die Hindernisse durch belegte und der Freiraum durch unbelegte Zellen repräsentiert werden. Aus Start- und Zielpose werden wie oben beschrieben relevante Bereiche ermittelt. Aus den Dimensionen dieser Bereiche und mit der Auflösung der Bewegungsprimitiven werden zur Laufzeit die Planungsumgebungen als Rasterkarten erstellt.

Wie beschrieben, soll in diesen Umgebungen für die Planung des Fahrspurwechsels ausschließlich die Freifläche des Vorgewendes genutzt werden. Daher werden für die Berechnung der Rasterkarte neben dem umschreibenden Rechteck die äußere und die innere Feldgrenze und gegebenenfalls vorhandene Hindernisse im Vorgewende benötigt.

Die Initialisierung des Bewegungsplaners mit der gewählten Zellauflösung, den gewählten Bewegungsprimitiven und dem im Verhältnis zur Zellauflösung sehr großen Grundrisses des Mähdreschers dauert auf dem Fahrzeugrechner mehrere Minuten und ist auf dem Feld daher nicht praktikabel. Daher wird ein gängiges Vorgehen aus der Robotik gewählt, bei dem alle Hindernisse um den halben Roboterradius vergrößert und der Roboter als punktförmig angenommen werden kann (siehe Abschnitt 3.1.3). Dem Planer wird entsprechend ein Grundriss von genau einer Zelle übergeben und die Feldgrenzen und Hindernisse werden geometrisch ausgedehnt. Die entstandenen Geometrien werden mit dem Rechteck verschnitten, das die Planungsumgebung beschreibt. Dies ist exemplarisch in Abbildung 5.21 dargestellt. Geometrisch ergibt sich der Freiraum $S_{frei}$ für die Umgebung $U$ mit der Fläche $O$ innerhalb der aufgeblähten äußeren und der Fläche $I$ innerhalb der aufgeblähten inneren Feldgrenze und den zusammengefassten

aufgeblähten Hindernisflächen $J$ als

$$S_{frei} = U \cap (O \setminus (I \cap J)). \tag{5.8}$$




(a) Die Feldgeometrie, die Start- und Zielpose, die Planungsumgebung und die aufgeblähten Feldgrenzen werden zur Berechnung des Freiraums benötigt.

(b) Die geometrisch berechneten Flächen für die Planungsumgebung, deren Differenz den Freiraum ergibt.

**Abbildung 5.21:** Berechnung des Freiraumes für die Planung im Vorgewende. Die freie Fläche berechnet sich geometrisch als die Fläche zwischen der aufgeblähten äußeren und aufgeblähten inneren Feldgrenze. Das entspricht in (b) der Differenz der grauen und der blau schraffierten Fläche für die Planungsumgebung.

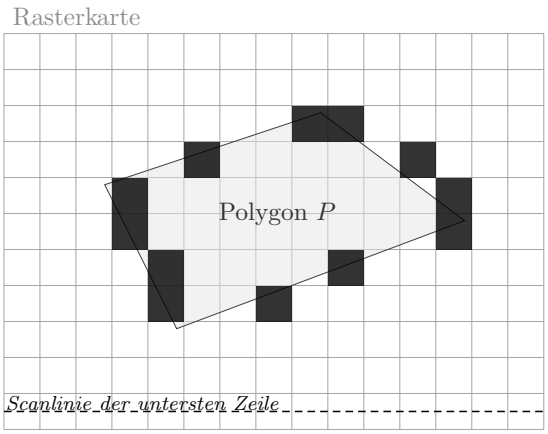
Die polygonale Darstellung des Freiraums muss in die Rasterkarte überführt werden. Dazu wird der in Algorithmus 6 aufgelistete Edge Flag Algorithmus [2] verwendet. Bei dem Edge Flag Algorithmus wird ein Polygon in zwei Schritten in ein Raster eingetragen. In dem ersten Schritt wird die Kontur des Polgyons in dem Raster repräsentiert. Dazu werden die Schnittpunkte aller Kanten mit den Scanlinien der Zeilen des Rasters bestimmt. Die erste Zelle rechts von einem der Schnittpunkte, deren Mittelpunkt rechts von dem Schnittpunkt liegt, wird auf den vorgegebenen Wert gesetzt. Ein Beispiel für die entstehende Kontur ist in Abbildung 5.22(a) gegeben. Der zweite Schritt dient dem Füllen des Polygons. Dazu wird ein Merker (Flag) eingeführt, der kodiert, ob die aktuelle Zelle einer betrachteten Zeile innerhalb des Polygons liegt oder nicht. Mit einer Fallunterscheidung anhand des Flags und des Wertes der betrachteten Zelle kann so die Kontur gefüllt werden. In Abbildung 5.22(b) ist exemplarisch ein Ergebnis des Algorithmus gegeben.

Für die Planung im Vorgewende wird die Umgebungskarte mit belegten Zellen initialisiert und die polygonale Freifläche wird mit dem Edge Flag Algorithmus als unbelegte Zellen eingetragen. Für den Fall einer einzelnen Umgebung für einen Wendevorgang aus Abbildung 5.21 sind die resultierende Rasterkarte und das Planungsergebnis für die Start- und Zielpose in Abbildung 5.23 dargestellt. Für den Fall, dass Start- und Zielpose in unterschiedlichen Planungsumgebungen

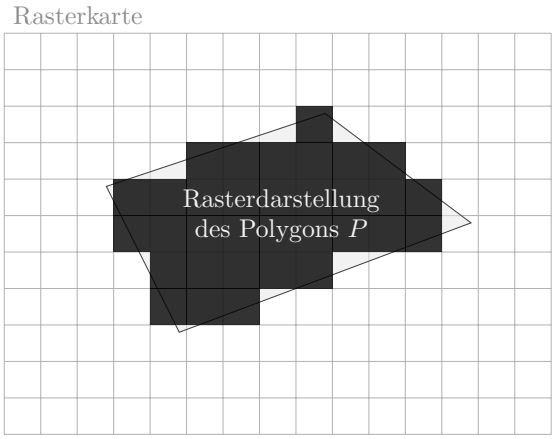
**Algorithmus 6** Edge Flag Algorithmus [2] zum Darstellen und Füllen eines Polygons $P$ mit dem Wert $v$ in einem Raster.

```
1: Prozedur EdgeFlag(P, v)
   A. Kontur eintragen
2:     für alle Kanten des Polygons P führe aus
3:         für alle Zeilen, die die Kante schneiden führe aus
4:             komplementiere linkesten Pixel, dessen Mittelpunkt rechts vom Schnittpunkt liegt
5:         Ende für
6:     Ende für
   B. Kontur füllen
7:     für alle Zeilen, die das Polygon P schneiden führe aus
8:         innerhalb ← FALSCH
9:         für x = 0 (links) bis x = x_max (rechts) führe aus
10:            falls Zeile[x] ist gesetzt dann
11:                negiere innerhalb
12:            Ende falls
13:            falls innerhalb ist WAHR dann
14:                setze Zeile[x] auf Füllwert v
15:            sonst
16:                setze Zeile[x] zurück
17:            Ende falls
18:        Ende für
19:    Ende für
20: Ende Prozedur
```

liegen, die durch eine Sollstrecke im Vorgewende verbunden werden, ist in Abbildung 5.24 ein Beispiel gegeben. In dem Bereich der Startpose wird mit dem gitterbasierten Planer der SBPL Bibliothek bis zum Start der Sollstrecke und im Bereich der Zielpose von dem Ende der Sollstrecke aus geplant. Die Sollstrecke wird mit dem in Abschnitt 5.2.2 vorgestellten Verfahren approximiert. Die erhaltenen Ergebnisse werden zu einem Pfad konkateniert, der das Wendemanöver für den Mähdrescher darstellt.




(a) Kontur des Polygons (dunkel eingefärbte Zellen) nach Schritt $A$ aus Algorithmus 6.




(b) Polygon in der Rasterdarstellung (dunkel eingefärbte Zellen) nach Schritt $B$ aus Algorithmus 6.

**Abbildung 5.22:** Beispiel des Edge Flag Algorithmus (nach [2]). In dem ersten Schritt wird die Kontur des Polygons eingetragen (a). Der zweite Schritt liefert das gefüllte Polygon in Rasterdarstellung (b).

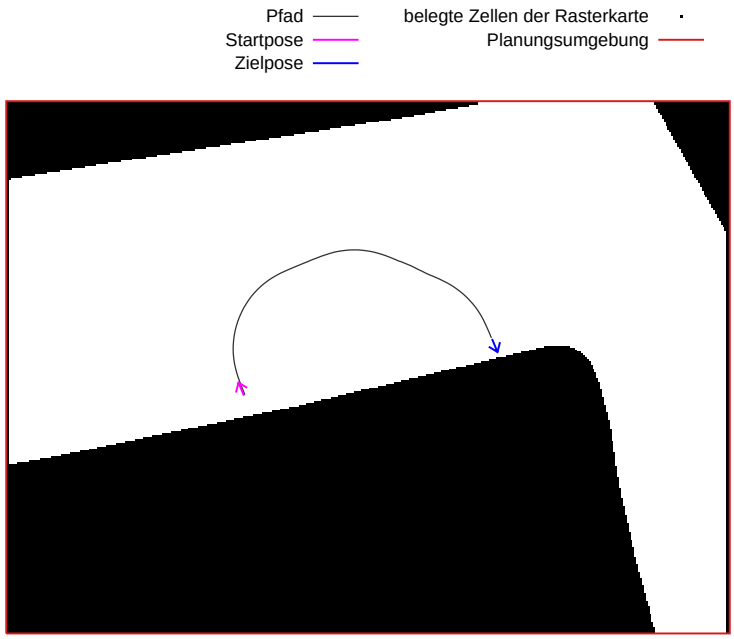



**Abbildung 5.23:** Rasterkarte der Planungsumgebung aus Abbildung 5.21 und Planungsergebnis des rasterbasierten Planers der SBPL Bibliothek.

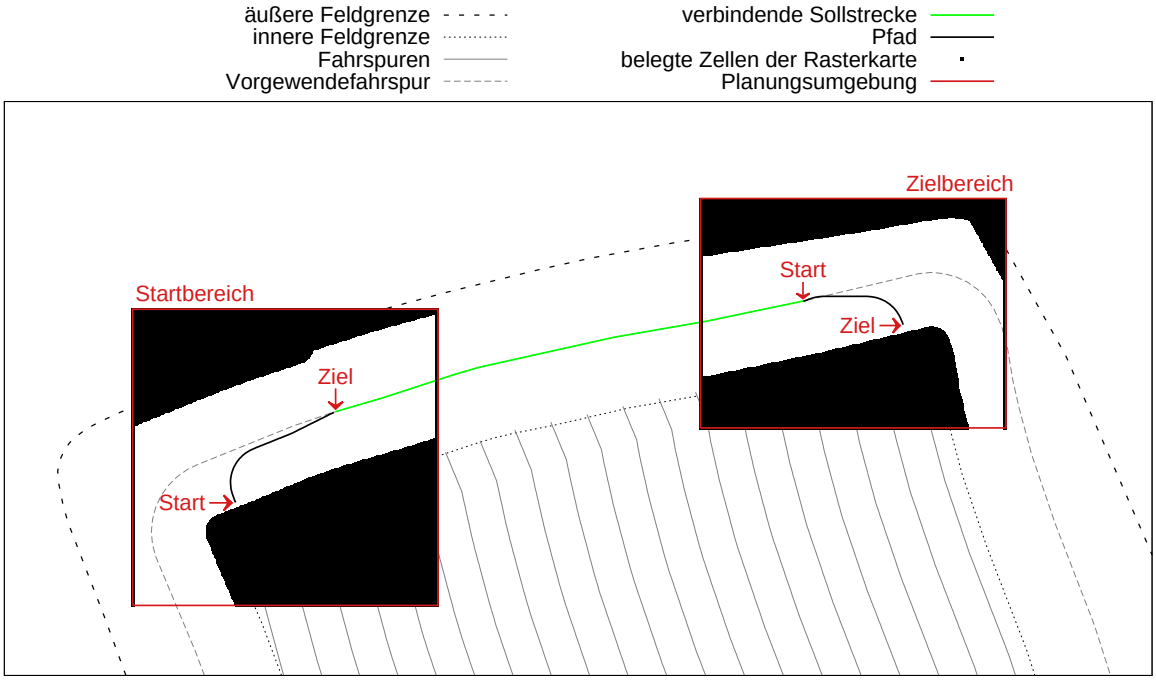
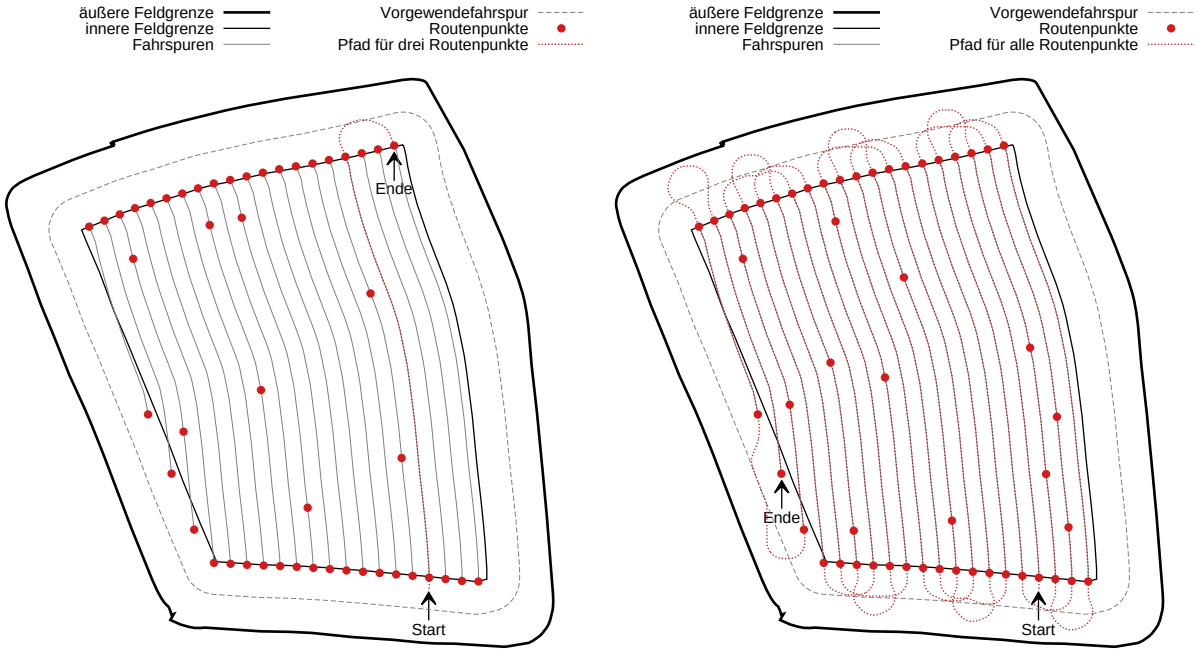



**Abbildung 5.24:** Wendemanöver des Mähdreschers mit unterschiedlichen Bereichen für Start- und Zielpose. In dem Start- und Zielbereich wird der rasterbasierte Planer der SBPL Bibliothek eingesetzt. Die Sollstrecke, die Start- und Zielbereich im Vorgewende verbindet, wird mit dem in Abschnitt 5.2.2 vorgestellten Verfahren approximiert. Die Verknüpfung der Teilpfade ergibt das Wendemanöver für den Mähdrescher.

### 5.3.5 Zusammenführen der Teilpfade

Es werden von der Bewegungsplanung immer die nächsten $n$ Routenpunkte mit den in Abschnitt 5.3.3 und Abschnitt 5.3.4 beschriebenen Verfahren aufgeplant. Die Anzahl der zu planenden Routenpunkte ist parametrierbar. In den Feldversuchen hat sich der Wert $n = 3$ bewährt. Beim Erreichen eines Routenpunkts und bei Abweichungen vom Pfad wird eine Neuplanung für die nächsten $n$ Routenpunkte angestoßen. Für die Teilpfade wird ein entsprechendes Geschwindigkeitsprofil erzeugt, so dass die Routenpunkte jeweils zum richtigen Zeitpunkt und mit der richtigen Geschwindigkeit erreicht werden. In Abbildung 5.25 sind exemplarisch die zusammengeführten Teilpfade als Planungsergebnis für die nächsten drei Routenpunkte und für die gesamte Route gezeigt.




(a) Planungsergebnis für die nächsten drei Routenpunkte des Mähdreschers.

(b) Planungsergebnis für die gesamte Route des Mähdreschers.

**Abbildung 5.25:** Zusammengeführte Teilpfade als Ergebnis der Bewegungsplanung für den Mähdrescher. Die Teilpfade werden nach den in Abschnitt 5.3.3 und Abschnitt 5.3.4 beschriebenen Verfahren erstellt. In (a) ist ein Planungsergebnis für die nächsten drei Routenpunkte, in (b) ein Planungsergebnis für die gesamte Route des Mähdreschers gezeigt.

Jeder Teilpfad ist von dem Mähdrescher abfahrbar und die Teilpfade werden nahtlos an den Routenpunkten zusammengeführt. Daher ist auch immer ein zusammengeführter Pfad, der die nächsten $n$ Routenpunkte verbindet, durch den Mähdrescher abfahrbar.

## 5.4 Bewegungsplanung für das Überladefahrzeug

Als Überladefahrzeug wurde in dem Projekt *marion* der Verbund eines Schleppers der Serie *CLAAS XERION* und einem Überladewagen der Firma *HAWE* eingesetzt (siehe Abschnitt 2.2). Die Kinematik des Überladefahrzeugs wird in Abschnitt 5.4.1 beschrieben. Für das Überladefahrzeug werden wie auch für den Mähdrescher die in Abschnitt 5.2 beschriebenen Verfahren zur Bewegungsplanung eingesetzt. Daher werden zur Planung ebenfalls Bewegungsprimitiven benötigt. Abschnitt 5.4.2 beschreibt die Bestimmung der Bewegungsprimitiven für das Überladefahrzeug.

Da das Überladefahrzeug sich innerhalb der inneren Feldgrenze nur auf abgeernteter Fläche bewegen darf, muss die noch nicht abgeerntete Feldfläche als dynamisches Hindernis für die Bewegungsplanung berücksichtigt werden. Im Gegensatz zum Mähdrescher darf das Überladefahrzeug außerdem die Fahrspuren mit erhöhten Kosten kreuzen. Daher wurde im Rahmen dieser Arbeit ein spezielles Verfahren entwickelt, mit dem die Bewegungsplanung eines Überladefahrzeugs für die Infield-Transportlogistik durchgeführt werden kann.

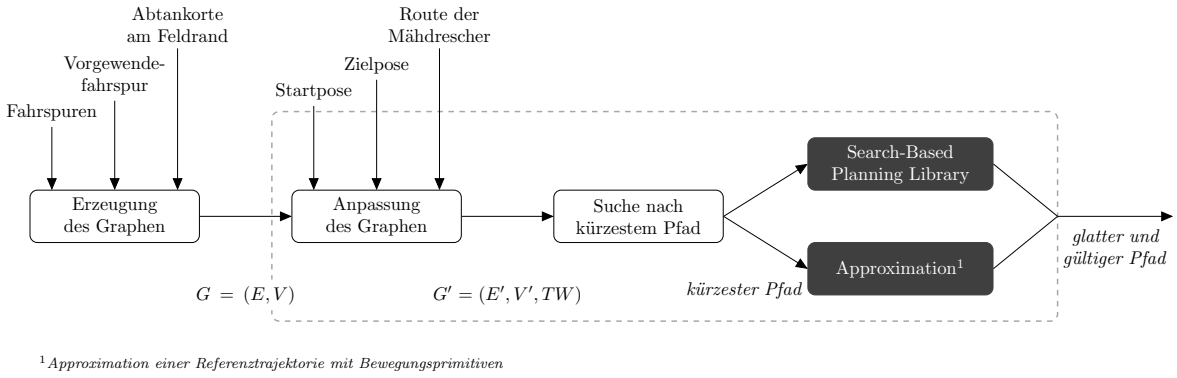



**Abbildung 5.26:** Übersicht der Bewegungsplanung für das Überladefahrzeug (angepasst von [85]). Aus der Feldgeometrie wird einmalig ein Graph erzeugt. Dieser wird für jeden Planungsprozess anhand von Start- und Zielpose und den aktuellen Routenplänen der Mähdrescher angepasst. In dem erzeugten Graphen wird anschließend der kürzeste Pfad gesucht. Dieser wird in Bereiche unterteilt, die mit einem Planer aus der SBPL und mit der Approximation einer Referenztrajektorie mit Bewegungsprimitiven abfahrbar gemacht werden. Durch das Zusammenführen der berechneten Teilpfade entsteht ein glatter und gültiger Pfad für das Überladefahrzeug.

Das Verfahren ist so ausgelegt, dass es auch für Prozesse mit mehreren Erntemaschinen geeignet ist. In Abbildung 5.26 sind die Schritte des Verfahrens dargestellt. Dabei wird aus den Fahrspuren, der Vorgewendefahrspur und den möglichen Abtankorten für das Überladefahrzeug am Feldrand ein gerichteter topologischer Graph erstellt. Die Erstellung des Graphen wird in Abschnitt 5.4.3 beschrieben. Dieser Graph wird für jede Planungsinstanz anhand der Routen der Erntefahrzeuge und der Start- und Zielpose des Überladefahrzeugs wie in Abschnitt 5.4.4 dargestellt angepasst. Die nicht abgeerntete Feldfläche wird dabei durch Zeitfenster an den Kanten repräsentiert. Wie in Abschnitt 5.4.5 beschrieben, kann in dem für die Planungsinstanz angepassten Graphen mit einer Graphsuche der kürzeste Weg unter Berücksichtigung der nicht

abgeernteten Feldfläche und des Startzeitpunktes bestimmt werden. Da dieser Pfad in dem topologischen Graphen nicht abfahrbar ist, wird er in Teilstücke zerlegt. Diese Teilstücke werden durch die in Abschnitt 5.2 beschriebenen Verfahren mit Hilfe der Bewegungsprimitiven in abfahrbare Teilpfade aufgeplant. Die Unterteilung wird in Abschnitt 5.4.6, die Aufplanung der Teile in Abschnitt 5.4.7 beziehungsweise in Abschnitt 5.4.8 beschrieben. Die Verknüpfung der Teilpfade aus Abschnitt 5.4.9 liefert den Gesamtpfad von der Start- zu der Zielpose für das Überladefahrzeug.

In der Literatur gibt es zur Bewegungsplanung von landwirtschaftlichen Transportfahrzeugen einige Arbeiten, beispielsweise [3, 14, 45] (siehe Abschnitt 3.1.6). Neu an dem in dieser Arbeit vorgestellten Verfahren ist die Modellierung des Feldes als dynamisches Hindernis durch Zeitfenster der Kanten eines topologischen Graphen und das Ermöglichen des Kreuzen von Fahrspuren aufgrund der Ausprägung des Graphen. Das vorgestellte Verfahren berücksichtigt in dem nachgeordneten Schritt der Erzeugung einer abfahrbaren Trajektorie aus dem topologischen kürzesten Pfad außerdem die kinematischen Einschränkungen des Überladefahrzeugs und ist für den Einsatz mit mehreren Erntefahrzeugen geeignet.

### 5.4.1 Kinematik des Überladefahrzeugs

Das kinematische Modell des Überladefahrzeugs ist in Abbildung 5.27 dargestellt. Der eingesetzte Schlepper verfügt über eine Allradlenkung. Der angehängte Überladewagen ist ein Tandemanhänger, so dass sein Fahrverhalten mit dem eines einachsigen Anhängers vergleichbar ist. In welcher Weise die Radstellungen des Schleppers durch das Lenksystem verstellt werden, war in dem Projekt *marion* a priori unbekannt, da das Lenksystem auch in diesem Fall durch einen Drittanbieter in der Projektlaufzeit entwickelt wurde. Daher ist für die Definition der Bewegungsprimitiven des Überladefahrzeugs eine experimentelle Bestimmung von Lenkparametern und -performanz durchgeführt worden.

### 5.4.2 Bestimmung von Bewegungsprimitiven

Wie im vorhergehenden Abschnitt beschrieben, müssen auch zur Bestimmung der Bewegungsprimitiven des Überladefahrzeugs Parameter und Performanz des eingesetzten Lenksystems experimentell bestimmt werden. Die Experimente dazu sind werden in Abschnitt 6.1 beschrieben. Die Generierung der Bewegungsprimitiven für das Überladefahrzeug aus diesen Parametern geschieht analog zu der in Abschnitt 5.3.2 beschriebenen Generierung der Bewegungsprimitiven für den Mähdrescher.

In Abbildung 5.28 sind die erstellten Sätze an Bewegungsprimitiven für das Überladefahrzeug gezeigt. Es werden auch für das Überladefahrzeug unterschiedliche Sätze an Bewegungsprimitiven generiert, weil für die Bewegungsplanung des Überladefahrzeugs auch die in Abschnitt 5.2 beschriebenen Verfahren mit unterschiedlichen angestrebten Fahrverhalten eingesetzt werden. In Abbildung 5.28(a) und Abbildung 5.28(b) sind die Bewegungsprimitiven für den SBPL Planer (Abschnitt 5.2.1), in Abbildung 5.28(c) und Abbildung 5.28(d) die Bewegungprimitiven für die

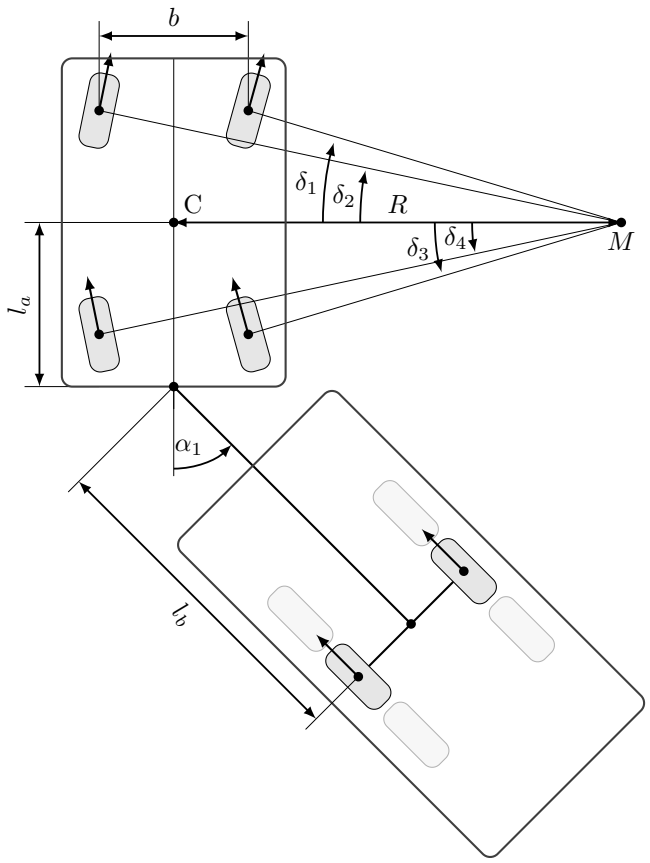
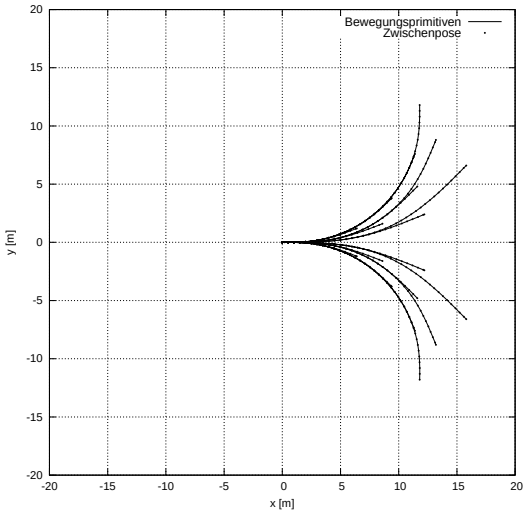


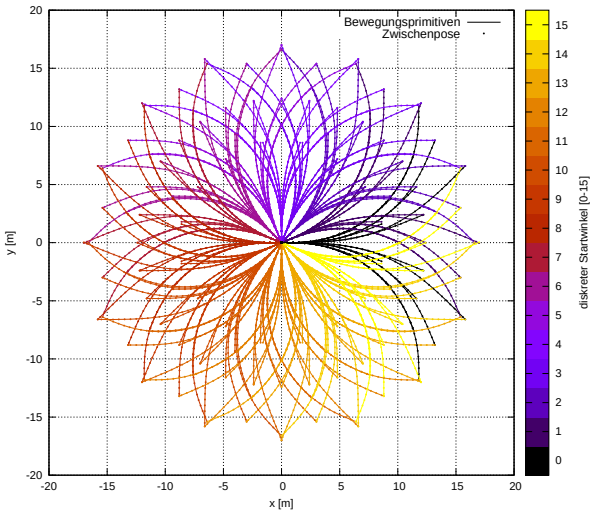

**Abbildung 5.27:** Kinematik des Überladefahrzeugs. Beide Achsen des Schleppers werden zur Lenkung genutzt. Der angehängte Überladewagen ist ein Tandemanhänger und verhält sich somit ähnlich wie ein einachsiger Anhänger. Das Überladefahrzeug verfügt insgesamt über eine komplexere Kinematik als der Mähdrescher (Abbildung 5.18).

Approximation einer Referenztrajektorie (Abschnitt 5.2.2) gegeben. Abbildung 5.28(a) und Abbildung 5.28(c) zeigen jeweils die Bewegungsprimitiven für den diskreten Startwinkel 0, während Abbildung 5.28(b) und Abbildung 5.28(d) den vollen Satz der Bewegungsprimitiven für alle 16 beziehungsweise 32 diskreten Startwinkel zeigt. Bei dem Vergleich von Abbildung 5.28(a) und Abbildung 5.28(c) wird ersichtlich, dass die Primitiven für den SBPL Planer weitere Kurvenfahrten beinhalten, während die Primitiven für die Approximation von Referenztrajektorien eher geradeaus fokussiert sind. Der Suchraum, der durch den Einsatz dieser Primitiven expandiert wird, ist in den beiden Fällen also unterschiedlich. Es sind grundsätzlich andere Fahrverhalten zu erwarten. Für die Approximation von Referenztrajektorien wird auch hier eine höhere Winkelauflösung gewählt, um genauere Ergebnisse erzielen zu können. Das ist für das Überladefahrzeug vor allem bei der Parallelfahrt, in der das Überladen in geringer seitlicher Distanz zum Mähdrescher durchgeführt wird, unbedingt erforderlich.

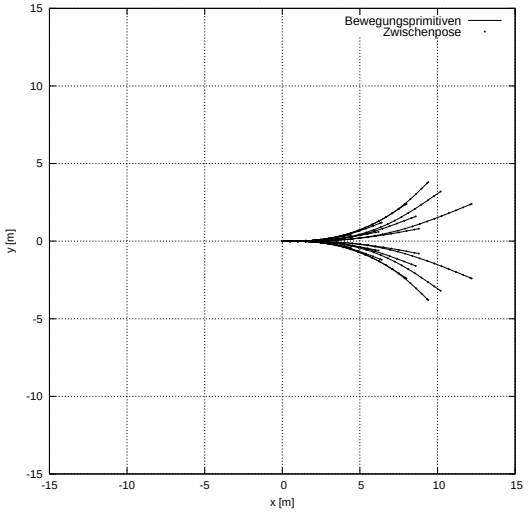
Im Vergleich zu den Bewegungsprimitiven des Mähdreschers aus Abbildung 5.19 werden größere



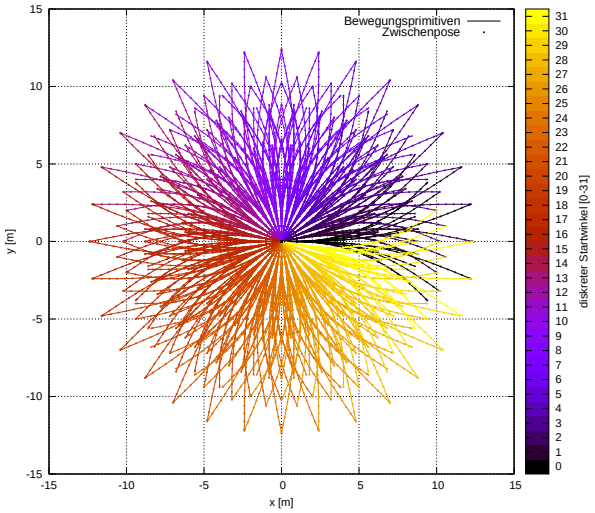

(a) Auszug der Bewegungprimitiven des Überladefahrzeugs für den SBPL Planer und den diskreten Startwinkel 0.




(b) Vollständiger Satz an Bewegungprimitiven des Überladefahrzeugs für den SBPL Planer mit allen 16 diskreten Startwinkeln.




(c) Auszug der Bewegungprimitiven des Überladefahrzeugs für die Approximation von Referenztrajektorien und den diskreten Startwinkel 0.



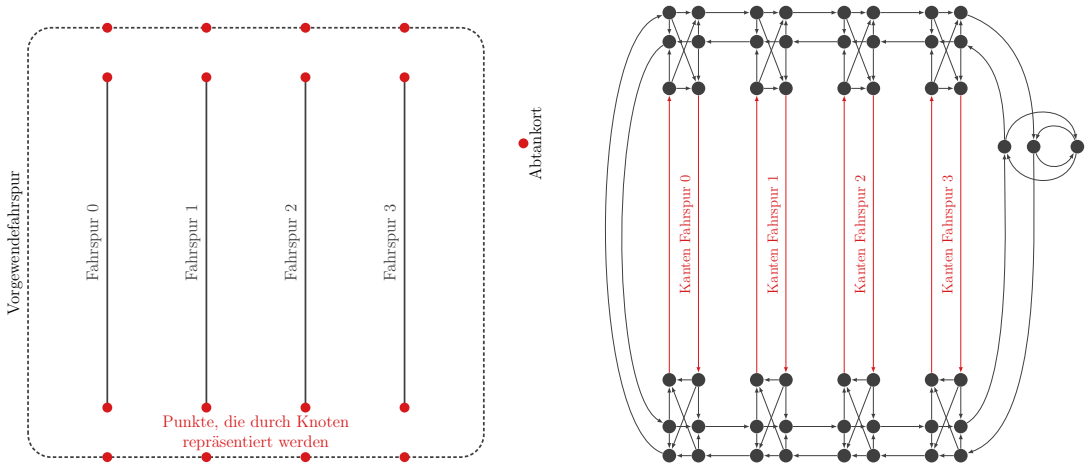

(d) Vollständiger Satz an Bewegungprimitiven des Überladefahrzeugs für die Approximation von Referenztrajektorien mit allen 16 diskreten Startwinkeln.

**Abbildung 5.28:** Die Bewegungsprimitiven des Überladefahrzeugs. Die zur Bewegungsplanung eingesetzten Verfahren aus Abschnitt 5.2 benötigen unterschiedlich ausgeprägte Bewegungprimitiven. In (a) und (b) sind die Bewegungsprimitiven für den SBPL Planer (Abschnitt 5.2.1), in (c) und (d) die Bewegungprimitiven für die Approximation einer Referenztrajektorie (Abschnitt 5.2.2) gegeben. (a) und (c) zeigen jeweils die Bewegungsprimitiven für den diskreten Startwinkel 0, während (b) und (d) den vollen Satz der Bewegungsprimitiven für alle 16 beziehungsweise 32 diskreten Startwinkel zeigt.

Radien vorgesehen. Das ist vor allem auf den Anhänger des Überladefahrzeugs zurückzuführen. Ähnlich zu dem Mähdrescher besteht hier die Randbedingung, dass das Überladefahrzeug ausschließlich vorwärts fahren soll. Dementsprechend sind nur vorwärts gerichtete Bewegungen in den Sätzen an Bewegungsprimitiven enthalten.

### 5.4.3 Repräsentation des Feldes in einem Graphen

Wie einleitend beschrieben, wird für die Bewegungsplanung des Überladefahrzeugs initial aus den Fahrspuren, der Vorgewendefahrspur und den Abtankorten am Feldrand ein topologischer gerichteter Graph erstellt. Dieser Graph ist Ausgangspunkt für jede Planungsinstanz. In Abbildung 5.29 ist schematisch an einem kleinen Beispiel mit vier Fahrspuren und einem Abtankort am Feldrand gezeigt, wie der Graph erstellt wird. Für jede Fahrspur werden Start- und Endpunkt jeweils durch zwei Knoten und eine gerichtete Kante repräsentiert. Die Knoten und Kanten der Fahrspur stellen unterschiedliche Fahrtrichtungen für das Überladefahrzeug dar (siehe Abbildung 5.29(b)). Das Gewicht der Fahrspurkanten entspricht der tatsächlichen Distanz der Fahrspur. Die Vorgewendefahrspur wird ebenfalls durch Knoten und Kanten in unterschiedlicher Fahrtrichtung repräsentiert. Die Verbindung von Fahrspuren und Vorgewendefahrspur entspricht der einer T-Kreuzung in Straßenkarten mit erlaubten Wendemanövern. Die Abtankorte am Feldrand werden ebenfalls in beide Fahrtrichtungen mit der Vorgewendefahrspur verbunden. In Abbildung 5.29(b) ist exemplarisch das Ergebnis der beschriebenen Repräsentation des Feldes aus Abbildung 5.29(a) in einem Graphen gegeben.



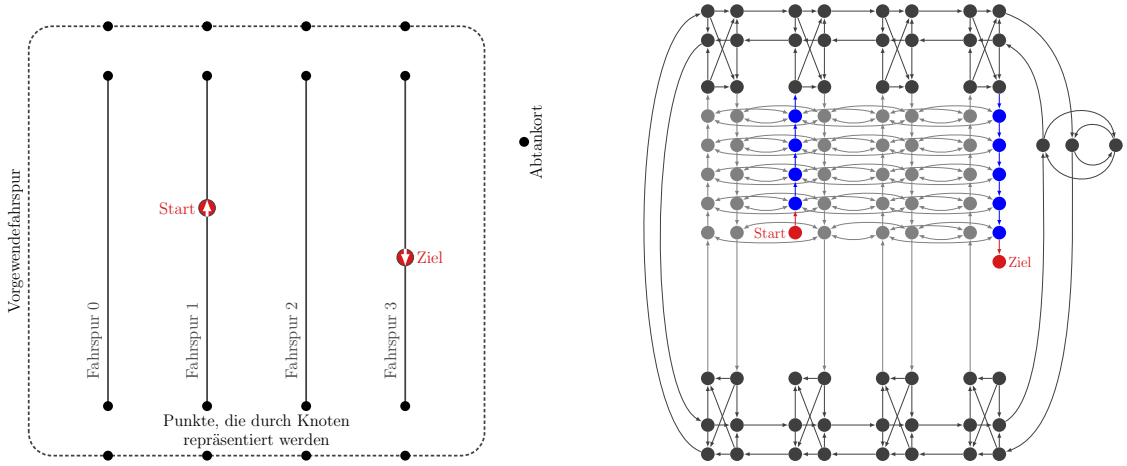

(a) Beispielgeometrie für ein Feld.

(b) Erzeugter Graph für die Feldgeometrie aus Abbildung 5.29(a).

**Abbildung 5.29:** Erzeugung eines gerichteten Graphen für die Bewegungsplanung des Überladefahrzeugs. Anhand der Fahrspuren, der Vorgewendefahrspur und den Feldzugangspunkten werden relevante Punkte extrahiert (a), die in dem erzeugten Graphen durch Knoten zu repräsentieren sind. Die Verbindung der Knoten liefert den gerichteten Graphen (b).

### 5.4.4 Dynamische Anpassung des Graphen

Für die Anpassung des Graphen werden der initiale topologische Graph aus Abschnitt 5.4.3, die Routen der Erntefahrzeuge und die Start- und Zielpose für das Überladefahrzeug benötigt. Es werden Knoten in den Graphen eingefügt, die die Start- und Zielpose repräsentieren. Liegt eine dieser Posen im Vorgewende, so wird der Knoten mit den nächsten Knoten der Vorgewendefahrspur verbunden. Liegt eine Pose auf einer Fahrspur, so sind mehrere Anpassungen nötig. In dem Beispiel aus Abbildung 5.30 liegen Start- und Zielpose jeweils auf einer Fahrspur.




(a) Probleminstanz mit Fahrspuren, Abtankort und Start- und Zielpose für das Überladefahrzeug.

(b) Dynamisch angepasster topologischer Graph für die Probleminstanz aus (a).

**Abbildung 5.30:** Dynamische Anpassung des topologischen Graphen. Start- und Zielpose für eine Planungsinstanz (a) werden als Knoten in den Graphen aus Abbildung 5.29(b) eingefügt. In diesem Fall liegen Start- und Zielpose jeweils auf einer Fahrspur. Entsprechend werden Zwischenknoten und -kanten eingefügt, wie in Abschnitt 5.4.4 beschrieben. Zusätzlich werden Kanten eingefügt, die Zwischenknoten benachbarter Fahrspuren orthogonal verbinden, um das Kreuzen von Fahrspuren zu ermöglichen. Für alle Kanten werden Zeitfenster berechnet. Diese Zeitfenster modellieren die jeweils durch eine Kante repräsentierte Feldfläche als dynamische Hindernis.

Wenn eine Startpose auf einer Fahrspur liegt, so wird die Fahrspurkante der aktuellen Fahrtrichtung entfernt und durch mehrere Kanten ersetzt. Dabei werden von dem Startknoten in einem parametrierbaren Abstand in der Fahrtrichtung auf der Fahrspur Zwischenknoten und -kanten eingefügt. Diese sind erforderlich, um die Auflösung der Diskretisierung durch den topologischen Graphen zu verfeinern. Durch diese Verfeinerung kann die nicht abgeerntete Fläche, die als Zeitfenster an den Kanten gespeichert wird, genauer repräsentiert werden. In der Anwendung bedeutet dies, dass ein Überladefahrzeug bei dem Einsatz mit mehreren Erntemaschinen in einem gewissen Abstand auf derselben Fahrspur hinter einem Erntefahrzeug her fahren kann, statt warten zu müssen, bis das Erntefahrzeug eine Fahrspur verlassen hat. Der erste eingefügte Knoten auf der Fahrspur nach dem Knoten der Startpose wird zusätzlich orthogonal mit benachbarten Fahrspuren verbunden, die wiederum orthogonal mit ihren benachbarten Fahrspuren verbunden werden. So wird das Kreuzen von Fahrspuren bei der Suche nach dem kürzesten Pfad

ermöglicht. Für die Gewichtung der orthogonalen Verbindungen wird die euklidische Distanz und ein parametrierbarer Multiplikator genutzt, mit dem die Erhöhung der Kosten beim Kreuzen der Fahrspuren eingestellt werden kann.

Wenn die Zielpose auf einer Fahrspur liegt, so wird analog zu dem Fall der Startpose auf einer Fahrspur vorgegangen. Der Unterschied ist das Einfügen der Zwischenknoten und -kanten auf der Fahrspur rückwärts zur Fahrtrichtung. Dieses Einfügen dient ebenfalls der feineren Auflösung der Diskretisierung durch den topologischen Graphen. Von dem eingefügten Knoten, der in Fahrtrichtung vor dem Knoten der Zielpose liegt, werden analog zur Startpose auf einer Fahrspur orthogonale Verbindungen zwischen benachbarten Fahrspuren propagiert. In Abbildung 5.30 ist schematisch ein Beispiel der Anpassung des topologischen Graphen gegeben.

Anhand der Routen der Erntefahrzeuge und den angegebenen Durchschnittsgeschwindigkeiten der Erntefahrzeuge werden für alle Kanten des angepassten Graphen die Zeiten berechnet, zu denen die Fahrspuren durch Bestand oder durch ein Erntefahrzeug belegt sind. Diese Zeiten werden als Zeitfenster an den Kanten des Graphen gespeichert und können so in der Suche nach dem kürzesten Pfad als dynamische Hindernisse berücksichtigt werden.

### 5.4.5 Bestimmung des kürzesten Weges

In dem angepassten Graphen kann mit gängigen Verfahren der kürzeste Pfad von dem Start- zu dem Zielknoten gesucht werden. Für die Bestimmung des kürzesten Pfades wird für das Überladefahrzeug der Dijkstra Algorithmus [26] verwendet und angepasst, so dass die Zeitfenster der Kanten berücksichtigt werden. Der Dijkstra Algorithmus wurde ausgewählt, da er die kürzesten Distanzen von einem Startknoten zu allen anderen Knoten bestimmt. So können beispielsweise die Distanzen zu allen Abtankorten am Feldrand in einem Schritt berechnet und der geeignetste Abtankort bestimmt werden.

In Algorithmus 7 ist der Dijkstra Algorithmus aufgelistet. Ausgangswerte für diesen sind ein Graph $G = (E, V)$ und ein Startknoten $v_{start}$. In den Listen *dist* und *pred* werden die Distanzen zu allen anderen Knoten des Graphen und dessen Vorgänger auf dem kürzesten Pfad bestimmt. Diese Listen werden mit den Werten $\infty$ für die Distanz und $-1$ für den Vorgängerknoten initialisiert. Eine Liste $Q$ aller Knoten wird für die Iteration angelegt. Die Distanz des Startknotens wird auf 0 gesetzt. Die Hauptschleife wird solange ausgeführt, bis in $Q$ keine Elemente mehr vorhanden sind. Dabei wird immer der Knoten mit der geringsten Distanz aus $Q$ als zu untersuchender Knoten $v_{nearest}$ ausgewählt. Dieser wird aus $Q$ entfernt, bevor alle Nachbarknoten von $v_{nearest}$, die in $Q$ enthalten sind, untersucht werden. Für alle diese Knoten $u$ wird untersucht, ob die Distanz $d$ nach $u$ über $v_{nearest}$ kürzer ist als die aktuell eingetragene Distanz $dist[u]$. Ist dies der Fall, so wird die Distanz $dist[u]$ aktualisiert und $v_{nearest}$ wird als der Vorgängerknoten $pred[u]$ gespeichert. Nachdem alle Knoten aus $Q$ untersucht wurden, sind die kürzesten Pfade zu allen Knoten bestimmt und in der Liste *dist* gespeichert. Diese Liste wird zurückgegeben. Der kürzeste Pfad von dem Startknoten $v_{start}$ zu einem beliebigen Knoten kann über die Vorgängerliste *pred* vom Zielknoten aus rückwärts rekonstruiert werden.

Für die Nutzung in diesem Projekt wurden Anpassungen vorgenommen, um Zeitfenster berück-

**Algorithmus 7** Dijkstra Algorithmus [26] zur Bestimmung der kürzesten Pfade in einem Graphen $G = (V, E)$ ausgehend von einem Startknoten $v_{start}$.

**Prozedur** DIJKSTRA($G, v_{start}$)
*A. Initialisierung*
    initialisiere Liste *dist* für Knotendistanzen
    initialisiere Liste *pred* für Vorgängerknoten
    initialisiere Liste $Q \leftarrow V$ für zu iterierende Knoten
    **für alle** Knoten $v \in V$ **führe aus**
        $dist[v] \leftarrow \infty$
        $pred[v] \leftarrow -1$
    **Ende für**
    $dist[v_{start}] \leftarrow 0$
*B. Kürzeste Pfade von $v_{start}$ zu allen Knoten bestimmen*
    **solange** $Q$ ist nicht leer **führe aus**
        finde Knoten $v_{nearest}$ aus $Q$ mit geringster Distanz $dist[v]$
        entferne $v_{nearest}$ aus $Q$
        **für alle** Nachbarn $u \in Q$ von $v_{nearest}$ **führe aus**
            berechne Distanz $d \leftarrow dist[u] + dist_{E(u,v)}$ nach $u$ über $v_{nearest}$
            **falls** $d < dist[u]$ **dann**
                $dist[u] \leftarrow d$
                $pred[u] \leftarrow v_{nearest}$
            **Ende falls**
        **Ende für**
    **Ende solange**
    **gib zurück** *dist*
**Ende Prozedur**

sichtigen zu können. In Algorithmus 8 ist der angepasste Dijkstra Algorithmus aufgelistet. Als Eingabewerte werden bei diesem zusätzlich der Startzeitpunkt $t_{start}$ und die Zeitfenster $W$ der Kanten benötigt. Die Zeitfenster bestehen jeweils aus einem Start- und einem Endzeitpunkt und beschreiben die Zeiträume, zu denen eine Kante durch Bestand oder durch ein Fahrzeug belegt ist. Durch diese Zeitfenster können so dynamische Hindernisse berücksichtigt werden. Als Distanzen werden in dem angepassten Algorithmus die Fahrzeiten zwischen den Knoten genutzt. Daher wird die Liste *dist* durch die Liste *time* ersetzt. Die Initialisierung erfolgt analog zum ursprünglichen Algorithmus, aber mit dem Unterschied, dass die Zeit für den Startknoten nicht auf 0, sondern auf den Startzeitpunkt $t_{start}$ gesetzt wird. In der Hauptschleife wird jeweils der Knoten $v_{nearest}$ aus $Q$ mit der geringsten Zeit untersucht. Für alle seine Nachbarknoten wird untersucht, ob sie über diesen Knoten schneller erreichbar sind. Dabei werden die Zeitfenster für die Kante $E(u, v)$ untersucht. Blockiert ein Zeitfenster die Kante zu dem aktuell geplanten Zeitpunkt, so wird eine Wartezeit hinzugerechnet. Falls die Zeit zum Nachbarknoten $u$ kürzer ist als die bisher eingetragene, so werden Zeit und Vorgänger in den Listen *time* und *pred* entsprechend aktualisiert. Nachdem alle Knoten untersucht wurden, wird die Liste *time* mit den kürzesten Zeiten zu allen Knoten zurückgegeben. Wie bei dem ursprünglichen Dijkstra Algorithmus kann

auch hier aus der Vorgängerliste der Pfad rekonstruiert werden.

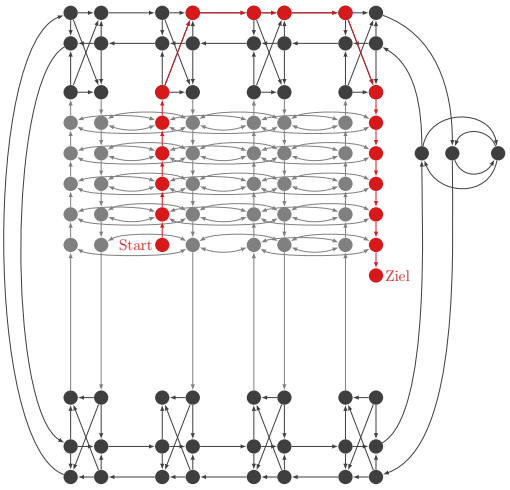
**Algorithmus 8** Dijkstra Algorithmus [26] zur Bestimmung der kürzesten Pfade in einem Graphen $G = (V, E)$ ausgehend von einem Startknoten $v_{start}$ und einem Startzeitpunkt $t_{start}$ unter Berücksichtigung der Zeitfenster $W$.

1: **Prozedur** DijkstraZeitfenster($G, v_{start}, t_{start}, W$)
*A. Initialisierung*
2: initialisiere Liste *time* für Knotenzeiten
3: initialisiere Liste *pred* für Vorgängerknoten
4: initialisiere Liste $Q \leftarrow V$ für zu iterierende Knoten
5: **für alle** Knoten $v \in V$ **führe aus**
6: $time[v] \leftarrow \infty$
7: $pred[v] \leftarrow -1$
8: **Ende für**
9: $time[v_{start}] \leftarrow t_{start}$
*B. Kürzeste Pfade von $v_{start}$ zu allen Knoten bestimmen*
10: **solange** $Q$ ist nicht leer **führe aus**
11: finde Knoten $v_{nearest}$ aus $Q$ mit geringster Zeit $time[v]$
12: entferne $v_{nearest}$ aus $Q$
13: **für alle** Nachbarn $u \in Q$ von $v_{nearest}$ **führe aus**
14: berechne Zeit $t \leftarrow time[u] + time_{E(u,v)}$ nach $u$ über $v_{nearest}$
15: setze Wartezeit $t_{wait} \leftarrow 0$
16: **für alle** Zeitfenster $w$ der Kante $E(u, v)$ **führe aus**
17: **falls** $time[u] > w_{start}$ **und** $time[u] < w_{end}$ **dann**
18: $t_{wait} \leftarrow w_{end} - time[u]$
19: **Ende falls**
20: **Ende für**
21: $t \leftarrow t + t_{wait}$
22: **falls** $t < time[u]$ **dann**
23: $time[u] \leftarrow t$
24: $pred[u] \leftarrow v_{nearest}$
25: **Ende falls**
26: **Ende für**
27: **Ende solange**
28: **gib zurück** *time*
29: **Ende Prozedur**

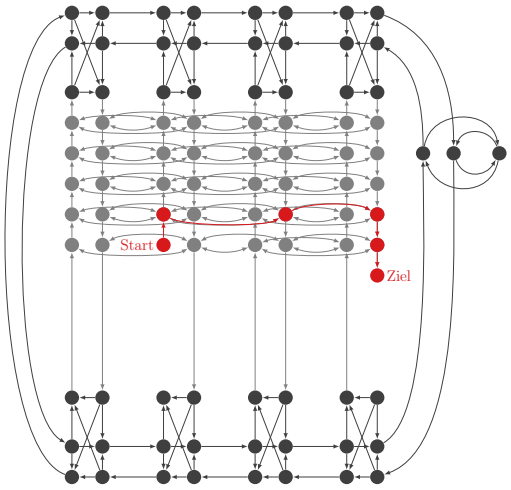
Alternativ zu dem Dijkstra Algorithmus könnte hier auch ein Algorithmus verwendet werden, der von einem Start- zu einem Zielknoten den kürzesten Pfad bestimmt. Eine solche Abfrage liefert beispielsweise der A*-Algorithmus, der eine bessere Performanz als der Dijkstra Algorithmus ermöglicht. Für die Bestimmung von kürzesten Wegen zu verschiedenen Zielknoten müsste ein solcher Algorithmus öfter ausgeführt werden.

Abbildung 5.31 zeigt exemplarisch Ergebnisse der Suche nach dem schnellsten Pfad mit dem an-

gepassten Dijkstra Algorithmus. Je nach Gewichtung der Kosten für das Kreuzen von Fahrspuren, den Routen der Erntefahrzeuge und dem Startzeitpunkt für den Bewegungsplan liefert der Algorithmus unterschiedliche Ergebnisse für dieselben Start- und Zielposen. In Abbildung 5.31(a) führt beispielsweise der rot dargestellte kürzeste Pfad durch das Vorgewende, während das Ergebnis aus Abbildung 5.31(b) die Fahrspuren kreuzt und das Vorgewende nicht nutzt.




(a) Beispielergebnis, bei dem der Pfad durch das Vorgewende führt.




(b) Beispielergebnis, bei dem Fahrspuren gekreuzt und das Vorgewende nicht besucht wird.

**Abbildung 5.31:** Beispielergebnisse für die Suche des kürzesten Weges (rot) in dem angepassten topologischen Graphen. In (a) ist exemplarisch ein Pfad gezeigt, der durch das Vorgewende führt, in (b) hingegen ein Pfad, der Fahrspuren kreuzt und den Fahrspurbereich nicht verlässt. Beide Ergebnisse haben ihre Berechtigung, denn das Ergebnis der Suche hängt von den Routen der Erntefahrzeuge, den zusätzlichen Kosten für das Kreuzen von Fahrspuren und von dem Startzeitpunkt ab.

In dem folgenden Abschnitt wird beschrieben, wie ein ermittelter Pfad in dem topologischen Graphen in unterschiedliche Bereiche unterteilt wird, die Fahrspurabschnitte oder Fahrspurwechsel repräsentieren.

### 5.4.6 Unterteilung des kürzesten Pfades

Da der kürzeste Pfad in dem topologischen Graphen nicht durch das Überladefahrzeug abfahrbar ist, muss dieser entsprechend verfeinert werden. Dazu wird der kürzeste Pfad in Abschnitte unterteilt, die auf derselben (Vorgewende-)Fahrspur liegen, und Abschnitte, die diese (Vorgewende-) Fahrspuren verbinden. Diese Abschnitte werden ähnlich wie bei der Bewegungsplanung für den Mähdrescher mit den in Abschnitt 5.2 beschriebenen Verfahren überplant. In Abbildung 5.32 ist diese Unterteilung für das Beispielergebnis aus Abbildung 5.31(a) gezeigt. Teilstücke des Pfades, die auf einer Fahrspur liegen, sind in rot dargestellt. Die Planung dieser Teile wird in Abschnitt 5.4.7 beschrieben. Blau dargestellt sind Teilstücke, die den Wechsel einer Fahrspur repräsentieren. Die Planung für diese Teile wird in Abschnitt 5.4.8 beschrieben.

Abschnitt 5.4.9 beschreibt das Zusammenfügen der geplanten Teilpfade zu einem durch das Überladefahrzeug ausführbaren Bewegungsplan von der Start- zu der Zielpose unter Berücksichtigung der dynamischen Hindernisse.

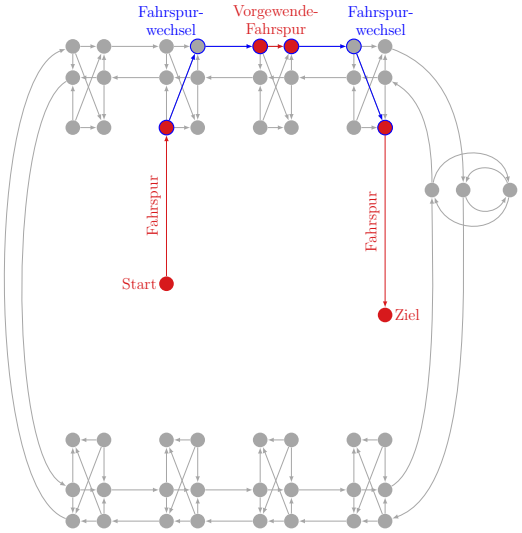



**Abbildung 5.32:** Unterteilung des kürzesten Pfades für das Beispielergebnis aus Abbildung 5.31(a) in Teile, die auf der Fahrspur liegen (rot) und Teile, die Fahrspurwechsel darstellen (blau). Zwecks Überschaubarkeit sind viele unbenötigte Knoten des angepassten topologischen Graphen ausgeblendet.

### 5.4.7 Planung für Abschnitte auf einer (Vorgewende-)Fahrspur

Die Abschnitte des ermittelten kürzesten Pfades in dem angepassten topologischen Graphen, die auf einer Fahrspur liegen, werden mit dem in Abschnitt 5.2.2 vorgestellten Verfahren zur Approximation einer Referenztrajektorie mit Bewegungsprimitiven aufgeplant. Das Ergebnis sind kinematisch gültige Teilpfade, die den jeweiligen, durch eine Knotenreihenfolge beschriebenen Abschnitt einer (Vorgewende-)Fahrspur approximieren. Das Vorgehen ist vergleichbar mit dem zur Bewegungsplanung des Mähdreschers aus Abschnitt 5.3.3. Neben dem Fahrspurabschnitt werden die Bewegungsprimitiven des Überladefahrzeugs und die Größe des Toleranzbereiches um die Referenztrajektorie benötigt. Auch hier repräsentiert der Toleranzbereich um die Referenztrajektorie den gültigen Suchraum, in dem ausgehend von der Startpose entlang der Referenztrajektorie eine möglichst gute Approximation bis zum Zielbereich gesucht wird (siehe Abschnitt 5.2.2).

Wie bei dem Mähdrescher geschieht bei der Planung mit dem Approximationsverfahren keine Überprüfung auf Hinderniskollisionen, da bei dem Verfahren angenommen wird, dass bei Einhaltung des Toleranzbereiches keine Kollisionen auftreten können. Abweichend zu der Planung auf einer Fahrspur für einen Mähdrescher muss jedoch bei der Planung für ein Überladefahrzeug die noch nicht abgeerntete Feldfläche als Hindernis berücksichtigt werden (Abschnitt 2.2). Daher wird eine Sollstrecke an jedem Punkt aufgeteilt, der durch einen Knoten mit Wartezei-

ten in dem topologischen kürzesten Pfad repräsentiert wird. Für die Wegpunkte der geplanten Teilpfade werden anschließend Geschwindigkeitsprofile berechnet, die eine Einhaltung der Wartezeiten und das Erreichen eines Knotens zu dem vorgegebenen Zeitpunkt und mit der passenden Geschwindigkeit gewährleisten. Falls dies beispielsweise durch die Überschreitung einer Maximalgeschwindigkeit nicht gewährleistet werden kann, so wird eine Fehlermeldung zurückgeliefert und die Planung wird unterbrochen. Dieser Fehler kann allerdings durch eine sorgfältige Wahl der Durchschnittsgeschwindigkeiten, die in der Suche im topologischen Graphen berücksichtigt werden, ausgeschlossen werden.

Die auf diese Weise erstellten Teilpfade approximieren die Fahrspurabschnitte eines topologischen Pfads und berücksichtigen dabei die noch nicht abgeerntete Feldfläche als dynamisches Hindernis für das Überladefahrzeug.

### 5.4.8 Planung für Verbindungen von (Vorgewende-)Fahrspuren

Für die Bereiche des kürzesten Pfades, die Fahrspurwechsel darstellen, wird der gitterbasierte Planer der SBPL eingesetzt (siehe Abschnitt 5.2.1). Dieser benötigt neben Start- und Zielpose und den Bewegungsprimitiven aus Abschnitt 5.4.2 eine Rasterkarte der Planungsumgebung, in der die durch Hindernisse belegte und die freie Fläche eingetragen sind. Diese Karte wird zur Laufzeit aus der Feldgeometrie und den an den Kanten des topologischen Graphen gespeicherten Zeitfenstern (siehe Abschnitt 5.4.4) erzeugt. Abbildung 5.33 zeigt schematisch das Vorgehen der Erzeugung der Hinderniskarte für die Planung eines Fahrspurwechsels. Für alle Fahrspuren, dessen Flächen die Planungsumgebung um die Start- und Zielpose schneiden, wird die zum Startzeitpunkt der Startpose noch nicht abgeerntete Fläche berechnet. Das Ergebnis der Verschneidung dieser noch nicht abgeernteten Fahrspurflächen (grau) und der Planungsumgebung (blau) wird in die diskrete Umgebungskarte als belegte Zellen eingetragen. In dieser Umgebung kann mit dem SBPL Planer ein Pfad von der Start- zu der Zielpose berechnet werden. Für den ermittelten Pfad und die gegebenen Eigenschaften der Start- und Endpose wird ein Geschwindigkeitsprofil erstellt, das zu einem Erreichen der Zielpose mit geforderter Geschwindigkeit zum geforderten Zeitpunkt führt.

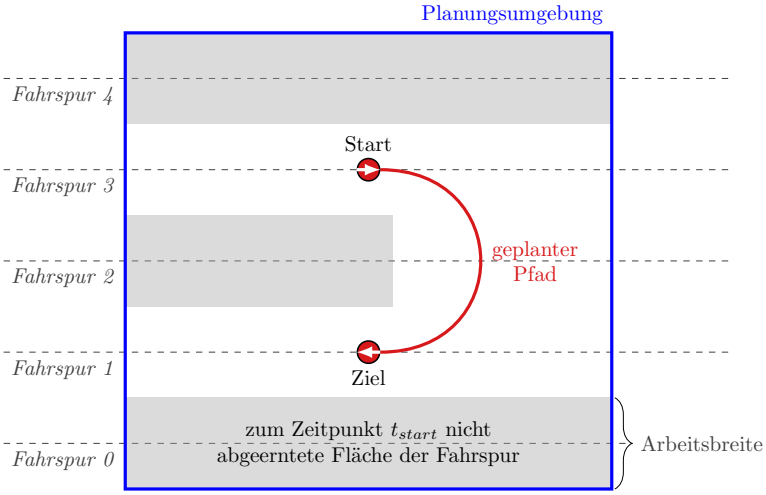


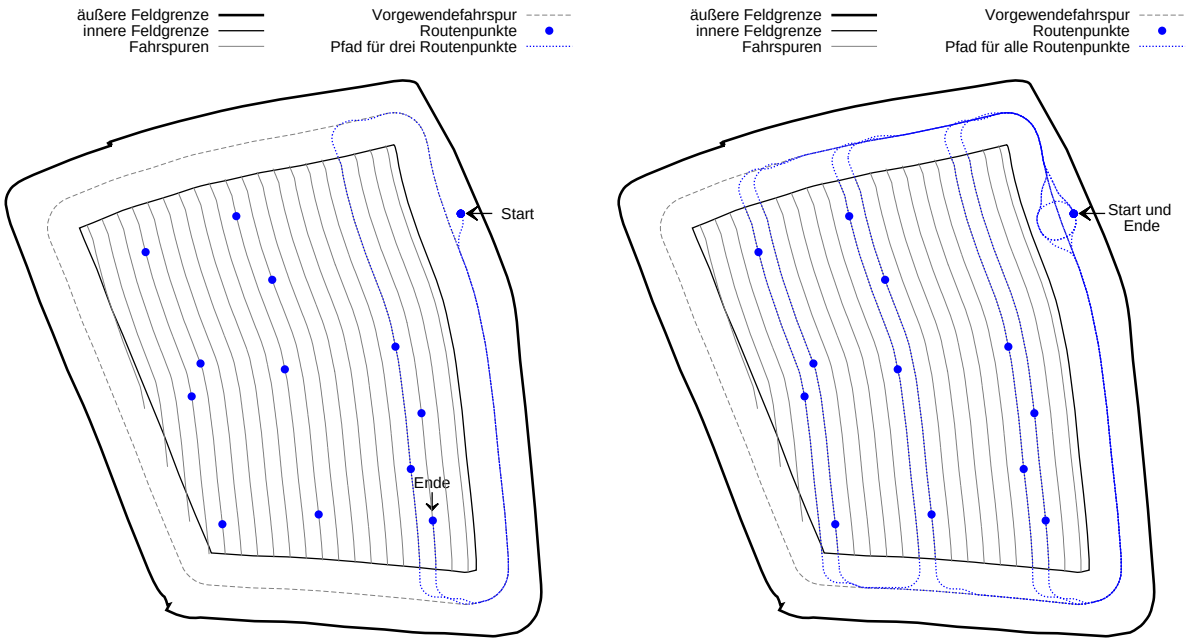

**Abbildung 5.33:** Schema der Planung für Verbindungen von (Vorgewende-)Fahrspuren des Überladefahrzeugs. Für die Planungsumgebung, in der die Start- und Zielpose liegen, wird die zum Startzeitpunkt $t_{start}$ noch nicht abgeerntete Fläche aller geschnittenen Fahrspuren als Hindernisfläche in die gerasterte Hinderniskarte eingetragen. Für die Planung des Pfades in der Umgebung wird der Planer der SBPL Bibliothek (Abschnitt 5.2.1) mit den in Abschnitt 5.4.2 beschriebenen Bewegungsprimitiven genutzt.

### 5.4.9 Zusammenführen der Teilpfade

Die mit den Verfahren aus Abschnitt 5.4.7 und Abschnitt 5.4.8 erstellten Teilpfade können nahtlos raumzeitlich zu einem Gesamtpfad von einer beliebigen Start- zu einer beliebigen Zielpose zusammengefügt werden. Ein auf diese Weise erstellter Pfad ist kinematisch gültig und berücksichtigt die abzuerntende Feldfläche als dynamisches Hindernis.

Wie auch bei der Bewegungsplanung für den Mähdrescher wird für das Überladefahrzeug während der Prozessausführung immer der Pfad für die nächsten drei Routenpunkte ausgehend von seiner aktuellen Pose berechnet. In Abbildung 5.34(a) ist das initiale Ergebnis für ein reales Feld aus einem Erntetest gegeben, bei dem das Überladefahrzeug den Prozess an dem Abtankort beginnt. Zur Vollständigkeit ist der Gesamtpfad des Überladefahrzeugs für die gesamte Route des Beispiels in Abbildung 5.34(b) gegeben. Im Vergleich zu dem Ergebnis der Bewegungsplanung desselben maschinenübergreifenden Routenplans für den Mähdrescher aus Abschnitt 5.25 werden die Aufgabenteilung und unterschiedlichen Fahrverhalten deutlich (Abschnitt 2.2). Das Überladefahrzeug befährt nur Fahrspuren, auf denen ein Überladeprozess durchgeführt wird und pendelt zwischen dem Mähdrescher und einem Abtankort am Feldrand, um das Erntegut von dem Mähdrescher zum Straßentransporter zu transportieren. Der Mähdrescher befährt jede Fahrspur, um die gesamte Feldfläche abzuernten. In dem Beispiel fährt das Überladefahrzeug den Abtankort am Feldrand drei mal an, nachdem der Mähdrescher jeweils mehrmals in Parallelfahrt abgetankt wurde.

Wie auch bei der Bewegungsplanung des Mähdreschers ist hier der aus mehreren Teilpfaden zusammengeführte Pfad für das Überladefahrzeug abfahrbar. Zusätzlich erlaubt die Bewegungs-




(a) Planungsergebnis für die nächsten drei Routenpunkte des Überladefahrzeugs. Diese Punkte beinhalten den ersten Überladeprozess und das Anfahren des Startpunkts für den zweiten Überladeprozess.

(b) Planungsergebnis für die gesamte Route des Überladefahrzeugs.

**Abbildung 5.34:** Zusammengeführte Teilpfade als Ergebnis der Bewegungsplanung für das Überladefahrzeug. Die Teilpfade werden nach den in Abschnitt 5.4.7 und Abschnitt 5.4.8 beschriebenen Verfahren erstellt. In (a) ist das Planungsergebnis für die nächsten drei Routenpunkte, in (b) das Planungsergebnis für die gesamte Route des Überladefahrzeugs.

planung des Überladefahrzeugs das Kreuzen von Fahrspuren mit erhöhten Kosten und berücksichtigt die abzuarbeitende Feldfläche als dynamisches Hindernis. Dies wird durch die den grundlegenden Verfahren der Bewegungsplanung vorgeschaltete Graphsuche ermöglicht.

# Kapitel 6

# Experimente und Ergebnisse

In diesem Kapitel werden Experimente zur Routen- und Bewegungsplanung mit den realen Maschinen beschrieben und die erzielten Ergebnisse diskutiert. In Abschnitt 6.1 werden die grundlegenden Lenkungstests zur Bestimmung von Lenkparametern und -performanzen beschrieben. Anschließend werden Experimente zur dynamischen Bewegungsplanung für beide Fahrzeugtypen in Abschnitt 6.2 und Experimente des gesamten Planungssystems in Abschnitt 6.3 erläutert.

## 6.1 Lenkungstests mit statischen Sollstrecken

Die zur Bewegungsplanung eingesetzten Verfahren (Abschnitt 5.2) benötigen einen Satz von Bewegungsprimitiven, von denen jede eine kinematisch gültige Bewegung abbildet. Wie in Abschnitt 5.1.2 beschrieben, wird bei der Planausführung beider Fahrzeugtypen die Regelung der Maschinenposition auf eine Sollstrecke durch ein Lenksystem realisiert. Um die Bewegungsprimitiven definieren zu können, sind der maximale Lenkwinkel und die maximale Lenkwinkeländerung, die von einem Lenksystem realisiert werden können, wichtige Parameter.

Da die Lenksysteme zugeliefert wurden und das jeweilige Lenkverhalten a priori nicht bekannt war, mussten diese Parameter experimentell bestimmt werden. Dazu werden Teststrecken mit unterschiedlichen maximalen Krümmungen und Krümmungsänderungen erstellt und in Experimenten durch das Lenksystem auf der realen Maschine abgefahren. In Abbildung 6.1 sind exemplarisch einige solcher Teststrecken dargestellt. Während des Abfahrens durch das Lenksystem werden die Maschinenpositionen aufgezeichnet. Anhand der Abweichung der aufgezeichneten Positionen von einer vorgegebenen Teststrecke können so die optimalen Werte bestimmt werden. Die Abweichung von einer Sollstrecke ist hier definiert als der Abstand einer Maschinenposition von dem dieser Position am nächsten liegenden Segments einer Sollstrecke. Sie entspricht somit der Länge der Lotstrecke der betrachteten Position auf ein Segment der vorgegebenen Sollstrecke, falls dessen Projektion auf ein Segment fällt, sonst dem euklidischen Abstand zu dem am nächsten liegenden Punkt des Segments.

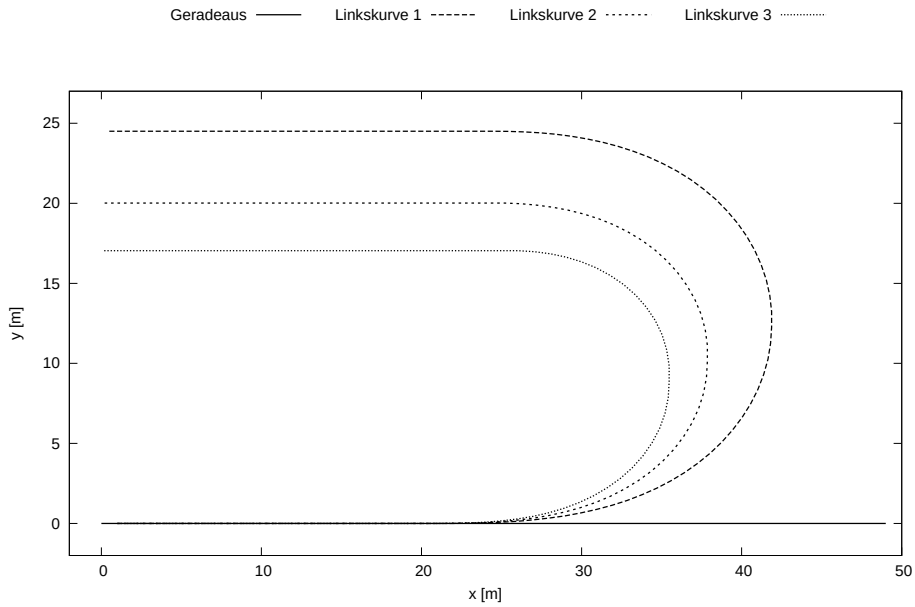



**Abbildung 6.1:** Beispiele für Teststrecken für die Bestimmung von Lenkparametern. Hier sind drei 180° Linkskurven mit unterschiedlicher Lenkwinkeländerung und gleichem maximalem Lenkwinkel und eine gerade Strecke dargestellt.

Für die Durchführung der Experimente wird die in Abbildung 6.2 gezeigte, zur Prozessausführung analoge Architektur (vergleiche Abbildung 5.2) verwendet. Die Bewegungsplanung wird durch eine Komponente ersetzt, die eine Teststrecke aus einer Datei einliest und eine Komponente, die gesendete Wegpunkte und sensorisch erfasste Posen für die Auswertung aufzeichnet. Die Schnittstellen der Komponenten und der Bewegungsplanung sind identisch. Die Teststrecke, die aus der Datei eingelesen wird, ist in einem lokalen kartesischen Koordinatensystem mit dem Startpunkt $(0, 0)$ und der Startorientierung 0° definiert. Da das Lenksystem Wegpunkte im Geokoordinatensystem WGS84 (siehe Abschnitt 3.2.1) erwartet, muss der Pfad vor dem Versenden transformiert werden. Dazu wird von dem Sender der Teststrecke zuerst die aktuelle

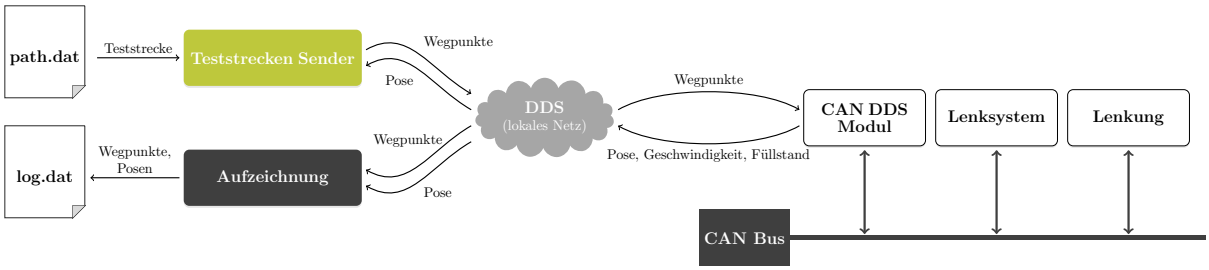
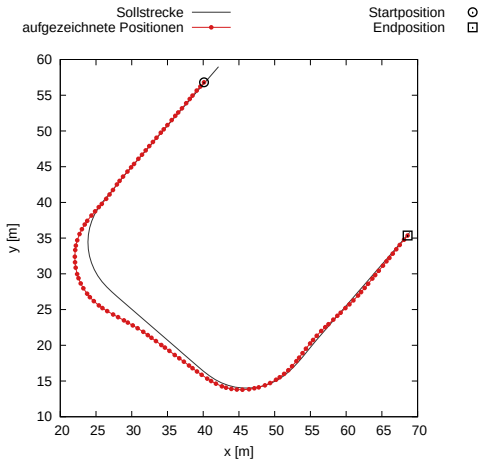


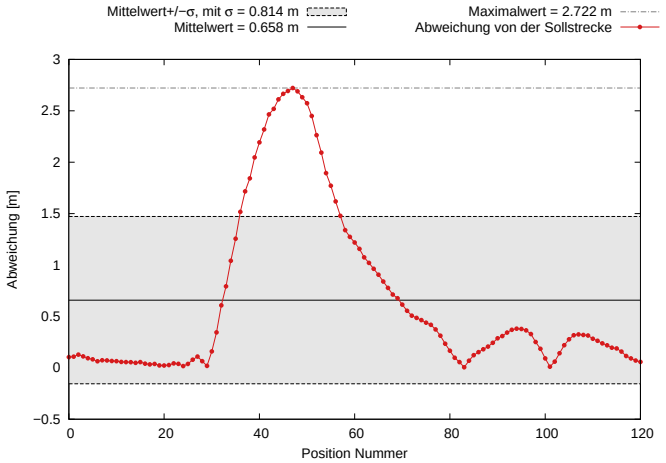

**Abbildung 6.2:** Architektur zur Durchführung von Experimenten zur Bestimmung von Bewegungsprimitiven und der Genauigkeit der Lenksysteme auf der jeweiligen Maschine (angepasst von [88]). Architektur und Schnittstellen zur Planausführung sind analog zu der Anbindung der Planausführung an die Bewegungsplanung (vergleiche Abbildung 5.2).

Pose des Mähdreschers abgerufen und in das lokale kartesische Koordinatensystem transformiert. Die Teststrecke wird in dem lokalen kartesischen Koordinatensystem an die aktuelle Pose des Mähdreschers rotiert und translatiert, in das Geokoordinatensystem rücktransformiert und anschließend per DDS dem Lenksystem in passender Form stückweise zur Verfügung gestellt (siehe Abschnitt 5.1.2).

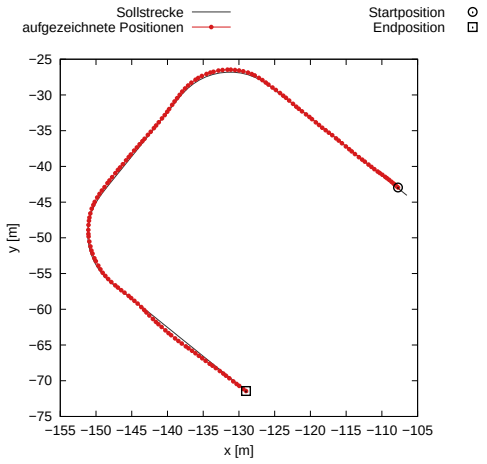
In Abbildung 6.3(a) ist exemplarisch ein Experiment mit ungültigen, in Abbildung 6.3(c) ein Experiment mit gültigen Werten für die Parameter dargestellt. Ergebnis dieser Experimente sind die Lenkparameter, mit denen Bewegungsprimitiven für den Mähdrescher (Abschnitt 5.3.2) und für das Überladefahrzeug (Abschnitt 5.4.2) erstellt werden können. Die erfolgreichen Tests verifizieren darüber hinaus die Schnittstelle zur Planausführung und zeigen generell, dass die Architektur zur Anbindung der Planausführung eine praktikable Lösung darstellt.



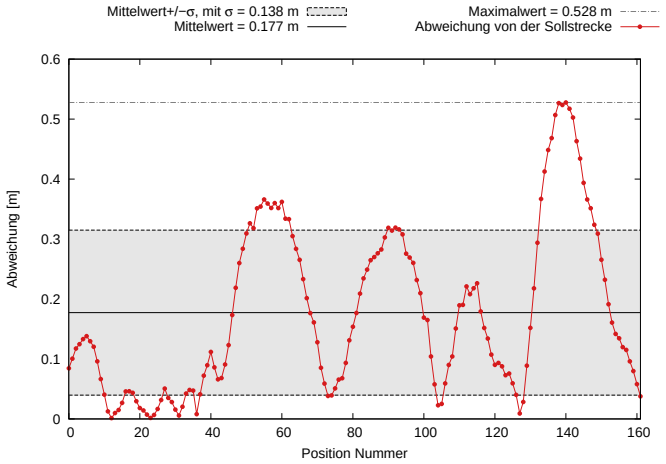

(a) Ungeeignete Teststrecke (schwarz) und aufgezeichnete Positionen (rot).




(b) Abweichungen der Positionen von der Teststrecke für (a).




(c) Geeignete Teststrecke (schwarz) und aufgezeichnete Positionen (rot).




(d) Abweichungen der Positionen von der Teststrecke für (c).

**Abbildung 6.3:** Beispiel der Experimente zur Bestimmung von Lenkparametern für den Mähdrescher. Für die Experimente wurden Teststrecken mit verschiedenen Krümmungen und Krümmungsänderungen generiert und mit dem Lenksystem abgefahren. Die Positionen wurden für die Auswertung aufgezeichnet. Exemplarisch ist in (a) das Ergebnis für eine ungeeignete, in (c) für eine geeignete Teststrecke gegeben.

## 6.2 Dynamische Bewegungsplanung

In den Experimenten zur dynamischen Bewegungsplanung werden die fahrzeugtypspezifischen Verfahren zur Bewegungsplanung getestet. Dazu werden statisch vorgeplante Routen in einer Datei hinterlegt und auf dem Fahrzeugrechner eingelesen. Für diese Routen werden die auf den realen Maschinen integrierten Verfahren zur Bewegungsplanung ausgeführt mit aktiviertem Lenksystem abgefahren. Die Auswertung der Experimente geschieht wie auch bei den Lenkungstests aus Abschnitt 6.1 über die Berechnung der Abstände aufgezeichneter Positionen zu einer Sollstrecke. Der Unterschied ist hier, dass die Sollstrecke nicht statisch vorgeplant und aus einer Datei eingelesen wird, sondern auf dem Fahrzeugrechner durch die entsprechenden Komponenten der Bewegungsplanung initial erzeugt und bei Bedarf neu berechnet wird.

Abbildung 6.4 zeigt das Ergebnis eines solchen Experiments für die dynamische Bewegungsplanung des Mähdreschers für ein Beet mit sechs Fahrspuren. Die schwarzen Punkte zeigen die während des autonomen Abfahrens aufgezeichneten Fahrzeugpositionen. Die von der dynamischen Bewegungsplanung berechneten und an das Lenksystem übertragenen Trajektorien sind als rot gestrichelte Linien abgebildet. In Abbildung 6.5 ist ein vergrößerter Ausschnitt gezeigt, in dem Abweichungen und entsprechend neu geplante Wendemanöver im Vorgewende zu erkennen sind.

Mit den Experimenten zur dynamischen Bewegungsplanung wird die Integration der in Abschnitt 5.3 und Abschnitt 5.4 vorgestellten Verfahren auf das jeweilige Fahrzeug getestet. Außerdem kann die Genauigkeit in der Planausführung für die auf den realen Maschinen mit den hinterlegten Bewegungsprimitiven generierten Strecken ermittelt werden. In dem Fall aus Abbildung 6.4 gibt es, wie allgemein in den Experimenten beobachtet, vor allem bei dem Wechsel von Fahrspuren Optimierungspotential. Möglicherweise trägt die in dem Projekt *marion* gegebene Schnittstelle der Übertragung von Wegpunkten mit jeweils einem Meter Abstand zueinander der ungenauen Kurvenfahrt bei. Es ist nicht bekannt, wie das Lenksystem die Wegpunkte interpoliert.

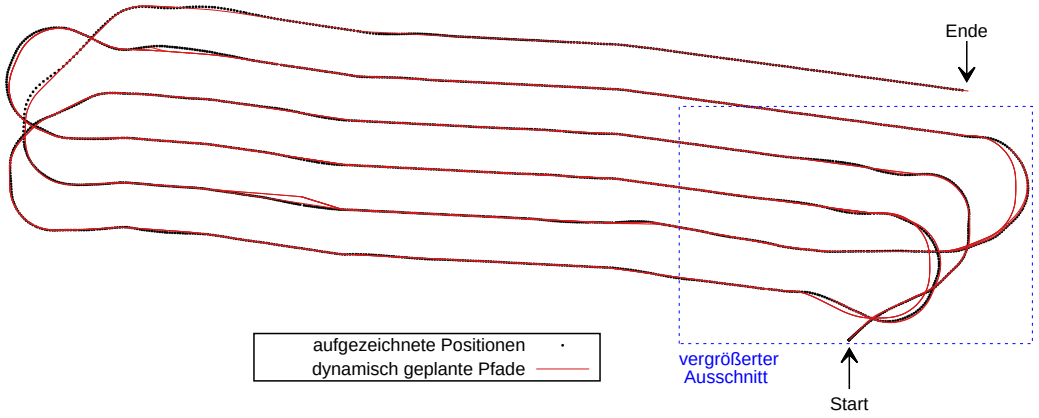



**Abbildung 6.4:** Ergebnisse eines Experiments zur dynamischen Bewegungsplanung des Mähdreschers. Für eine statische Route eines Beets mit sechs Fahrspuren wurde die dynamische Bewegungsplanung auf dem Fahrzeug und während der Planausführung berechnet. Die berechneten Trajektorien sind rot, die aufgezeichneten Maschinenpostionen schwarz eingezeichnet. Es ist erkennbar, dass die Abweichungen im Vorgewende wesentlich größer sind als die Abweichungen auf einer Fahrspur. Daher wird vor allem bei Wendemanövern eine dynamische Neuplanung angestoßen, während das auf der Fahrspur nur in seltenen Fällen erforderlich ist.

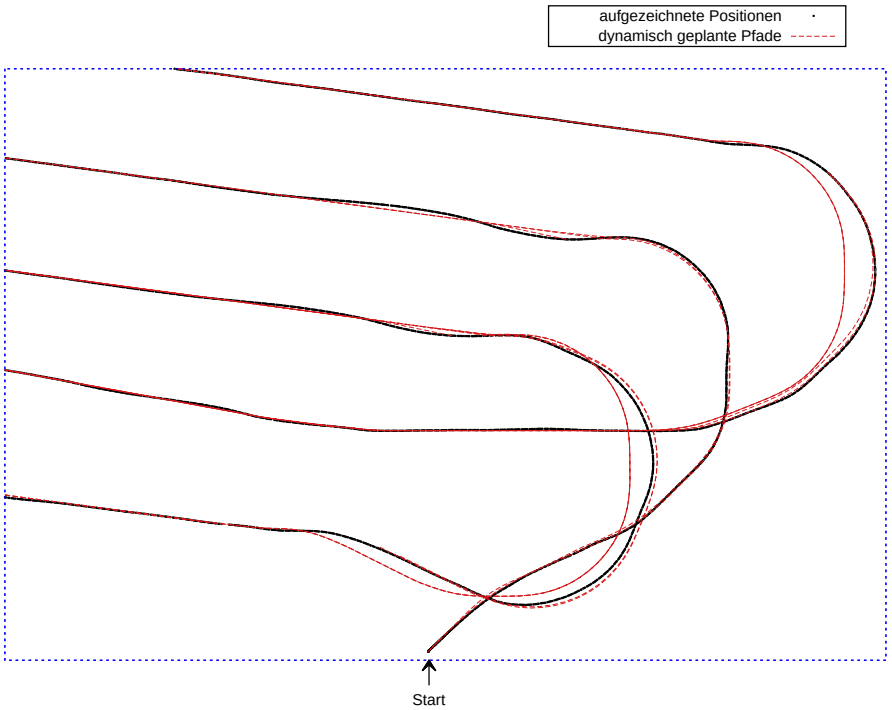



**Abbildung 6.5:** Ausschnitt des Experiments zur dynamischen Bewegungsplanung aus Abbildung 6.4. Bei Wendemanövern treten in diesem Experiment Abweichungen auf, die oft zu dynamischen Plananpassungen führen.

## 6.3 Experimente des gesamten Planungssystems

In Rahmen des Projekts *marion* konnte das entwickelte Planungssystem für einen Mähdrescher und ein Überladefahrzeug in realen Ernteprozessen erfolgreich getestet werden. In Abbildung 6.6 ist exemplarisch ein Ergebnis des Planungssystems für ein Feld aus den Erntetests gegeben.

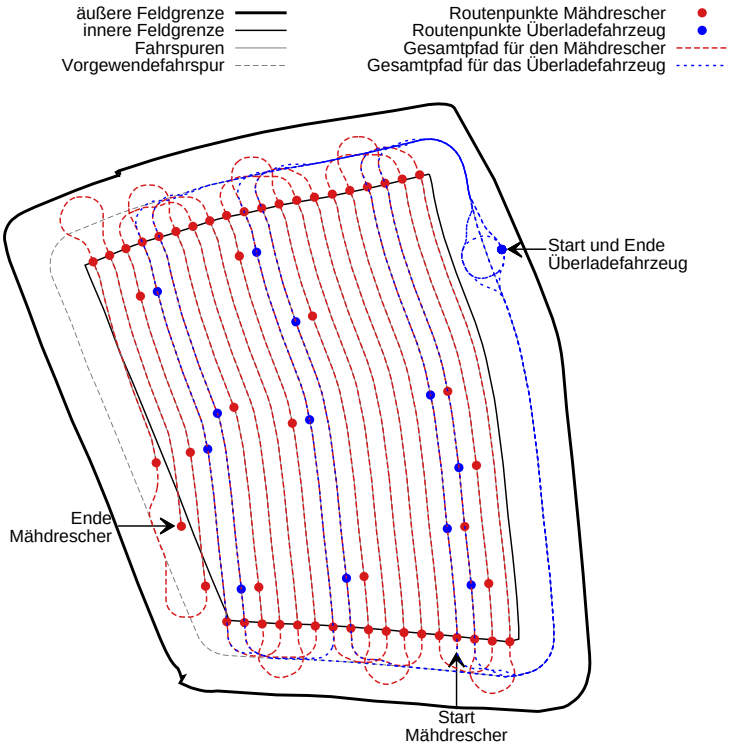



**Abbildung 6.6:** Beispielergebnis des Planungssystems für eine Feldgeometrie aus den realen Erntetests. Die Route und der geplante Pfad für den Gesamtprozess sind für den Mähdrescher in rot, für das Überladefahrzeug in blau dargestellt. Die Pläne der beiden Fahrzeuge sind einzeln in Abbildung 5.25 für den Mähdrescher und in Abbildung 5.34 für das Überladefahrzeug gegeben.

Ausgehend von der Aufzeichnung der Feldgeometrie konnten Referenzlinien erkannt, Fahrspuren generiert, ein maschinenübergreifender Gesamtplan erstellt, maschinenspezifisch verfeinert, dynamisch angepasst und ausgeführt werden. Die Planung und Einhaltung raumzeitlicher Treffpunkte und das Überladen in Parallelfahrt konnten dabei erfolgreich getestet werden. Bei vorhandenem GPS mit Korrektursignal wurden die Fahrspuren dabei durch den Mähdrescher so genau abgefahren, dass der gesamte Bestand flächendeckend abgeerntet wurde. Somit konnten in realen Erntetests die Funktionalität und Anwendbarkeit des Konzepts des Gesamtsystems sowie der entwickelten Verfahren der verschiedenen Planungsebenen gezeigt werden.

# Kapitel 7

# Zusammenfassung und Ausblick

Im Rahmen dieser Arbeit wurde ein Planungssystem für das Szenario der kooperativen Getreideernte entwickelt. Das Planungssystem berechnet für ein Ernte- und ein Transportfahrzeug einen gemeinsamen Prozessplan. Dieser Plan beschreibt den Ernte- und den Transportprozess des Ernteguts. Wie in der Zielsetzung formuliert, wird bei der Planung ausschließlich die Infield-Transportlogistik, also der Transport des Ernteguts bis zum Feldrand, betrachtet.

Dazu wurde in dieser Arbeit das Szenario der Infield-Transportlogistik und das aus diesem Szenario hervorgehende raumzeitliche Planungsproblem ausführlich beschrieben und formal dargestellt. Basierend auf der Formalisierung und unter Berücksichtigung der technischen Randbedingungen wurde das Konzept und die Implementierung eines raumzeitlichen Planungsverfahrens beschrieben. Das Konzept des Planungssystems sieht eine Aufteilung in maschinenübergreifende Routen- und maschinenspezifische Bewegungsplanung vor.

Weil für die beiden Fahrzeugtypen unterschiedliche Aufgaben und Randbedingungen existieren, wurden fahrzeugtypspezifische Verfahren zur Bewegungsplanung entwickelt. Diese Verfahren nutzen jeweils den gitterbasierten Planer der *Search-Based Planning Library* und die in dieser Arbeit entwickelte Approximation einer Referenztrajektorie zur Bewegungsplanung. Für diese Algorithmen wird eine Beschreibung der kinematischen Eigenschaften der Fahrzeuge in Form von Bewegungsprimitiven benötigt. Diese wurden im Rahmen dieser Arbeit mit Experimenten der realen Maschinen bestimmt und verifiziert. Die implementierten Verfahren zur Bewegungsplanung wurden auf Laborrechnern und mit realen Fahrzeugen erfolgreich getestet.

Durch eine zentrale Routenplanung wird sichergestellt, dass alle Fahrzeuge immer demselben gemeinsamen Prozessplan folgen. Die Routenplanung realisiert die anvisierte Optimierung des Gesamtprozesses anhand einer austauschbaren Kostenfunktion. Die fahrzeugspezifischen Teilpläne verfeinern immer denselben gemeinsamen übergeordneten Plan für das jeweilige Fahrzeug. Kann der gemeinsame Prozessplan von einem Fahrzeug nicht erfüllt werden, so wird dieser zentral überplant und an alle Fahrzeuge verteilt. Dadurch wird die geforderte Konsistenz der Teilpläne aller Fahrzeuge sichergestellt. Die Trennung der Routen- und der Bewegungsplanung ermöglicht außerdem eine Reduktion der Komplexität der einzelnen Planungsschritte. Die da-

durch ermöglichte Performanz erlaubt die dynamische Neuplanung auf den Fahrzeugrechnern bei Abweichungen in der Prozessausführung.

Die in Abschnitt 1.1 beschriebenen Zielsetzungen dieser Arbeit wurden erreicht:

**Formalisierung:** In Abschnitt 2.3 wurde die aus dem landwirtschaftlichen Szenario hervorgehende Planungsaufgabe formal beschrieben. Darüber hinaus wurden für den gewählten Ansatz die Teilaufgaben der Planung von Fahrspurreihenfolgen der Erntefahrzeuge (Abschnitt 4.3), von Überladeprozessen (Abschnitt 4.4) und die Aufplanung der Routen zu abfahrbaren Trajektorien des Mähdreschers (Abschnitt 5.3) und des Überladefahrzeugs (Abschnitt 5.4) detailliert dargestellt. Mit Hilfe dieser Formulierungen konnten die beschriebenen Planungsverfahren entwickelt werden.

**Raumzeitliche Planung:** Mit der in Kapitel 4 beschriebenen Routenplanung wurde ein Verfahren zur raumzeitlichen Planung von kooperativen Ernteprozessen mit einem Mähdrescher und einem Überladefahrzeug vorgestellt. Die Planung der kooperativen Überladeprozesse geschieht in einer topologischen Graphdarstellung der Probleminstanz und wird mit der in Abschnitt 4.4 beschriebenen impliziten Baumsuche durchgeführt.

**Prozessoptimierung:** Die in Kapitel 4 beschriebene Routenplanung ermöglicht eine Optimierung der Kooperation der beteiligten Maschinen bezüglich einer Optimierungsfunktion. Das aus Prozesssicht optimale Ergebnis für ein reales Szenario ist ein Kompromiss zwischen Stabilität und Optimalität eines Routenplans. Wie in Abschnitt 4.6 diskutiert, wird dies in dem vorgestellten Ansatz mit den eingesetzten Beetstrategien und der Optimierung der Überladevorgänge erreicht.

**Dynamische Plananpassung:** Eine dynamische Plananpassung wurde sowohl für die maschinenübergreifende Routenplanung als auch für die maschinenspezifische Bewegungsplanung umgesetzt (Abschnitt 5.1.3). Mit Hilfe dieser dynamischen Plananpassungen kann auf Abweichungen in der Prozessausführung reagiert werden.

**Konsistenz der Teilpläne:** Durch die gewählte Architektur mit einer zentralen Routenplanung (siehe Abschnitt 3.2.3) und dadurch, dass nur bei bestehender Funkverbindung neue Routenpläne generiert und verteilt werden, gibt es keine konkurrierenden Routenpläne. Weil jeder Teilplan einer Maschine dem aktuellen Routenplan folgen muss, sind diese ebenfalls konsistent.

**Integration und Tests:** Das Planungssystem wurde auf realen Maschinen integriert und konnte in Ernteprozessen erfolgreich getestet werden (siehe Abschnitt 6.3).

Es wurde somit gezeigt, dass die entwickelten Verfahren für die dynamische raumzeitliche Planung in dem Szenario der Infield-Transportlogistik geeignet sind. Als mögliche Punkte für die zukünftige Forschung und Weiterentwicklung bieten sich folgende offene Probleme an:

**Evaluation des Planungssystems:** Im Rahmen des Projekts *marion* konnte das entwickelte Planungssystem zwar in Ernteprozessen getestet, aber nicht evaluiert werden. Ohne entsprechende raumzeitliche Aufzeichnungen konnte kein Vergleich zwischen Prozessausführungen mit und ohne Planungssystem erfolgen. Solche Aufzeichnungen für die gleichen

Felder mit möglichst ähnlichen Umwelt- und Prozessbedingungen könnten über mehrere Jahre gesammelt einen Aufschluss über eine mögliche Effizienzsteigerung durch das Planungssystem geben. Eine solche Evaluation gestaltet sich jedoch schwierig aufgrund des hohen Einflusses der Umweltbedingungen auf den Ernteprozess, beispielsweise in Form von Ertrag, Feuchte, Bodenbeschaffenheit und Wetter. Aufschlussreich wäre aber sicherlich der Vergleich von Ausfallzeiten, die durch einen vollen Korntank eines Mähdreschers hervorgerufen werden. Anhand des Vergleichs könnte gezeigt werden, ob das Planungssystem diese teuren Standzeiten wie vorgesehen minimiert.

**Integrierte Abarbeitungs- und Überladeplanung:** Das vorgestellte Verfahren zur raumzeitlichen Routenplanung zur Optimierung des Gesamtprozesses ist unterteilt in die Bestimmung der Fahrspurreihenfolge des Mähdreschers und die Überladeplanung beider Fahrzeugtypen. Die Planung der Fahrspurreihenfolge ist dabei Eingabe für die Überladeplanung. Die beiden zugrunde liegenden Planungsprobleme sind Optimierungsprobleme, die sich gegenseitig raumzeitlich beeinflussen und die potentiell gegenläufige Ziele verfolgen. Während für die Planung der Fahrspurreihenfolge die Minimierung der Prozesszeit und Fahrstrecke anstrebt wird, ist das Ziel der Überladeplanung das Verhindern von Unterbrechungen in der Prozessausführung durch vollständig gefüllte Korntanks. In der beschriebenen Umsetzung der Routenplanung werden die Fahrspurreihenfolgen durch in der Praxis angewandte Beetstrategien festgelegt. Diese Strategien erlauben das Entleeren des Mähdreschers in Parallelfahrt zu einem Überladefahrzeug auf möglichst vielen Fahrspuren und sorgen somit für einen stabilen Prozess. Im Allgemeinen resultieren die Strategien jedoch in zusätzlich zurückzulegenden Distanzen im Vorgewende. Da das Planungssystem die Überladeprozesse prognostiziert, könnten diese fixen Strategien durch ein Optimierungsverfahren ersetzt werden, das nur zu den nötigen Zeitpunkten die Möglichkeit zum Überladen garantiert. Dabei müsste die Überladeplanung in das Verfahren integriert werden, um ungültige Lösungen auszuschließen. Eine solche integrierte Abarbeitungs- und Überladeplanung könnte die Effizienz der Ernteprozesse steigern. Aufgrund der in dieser Arbeit beschriebenen Komplexität der Überladeplanung ist davon auszugehen, dass eine praktikable Laufzeit auf Fahrzeugrechnern eines solchen Verfahrens eine große Herausforderung darstellt.

**Anbindung der außerfeldlichen Transportlogistik:** In dieser Arbeit wurde ausschließlich die Logistik des Ernteguts bis zum Feldrand, die sogenannte Infield-Transportlogistik betrachtet. Für reale Ernteprozesse besteht die Logistikkette jedoch zusätzlich aus der außerfeldlichen Abfuhr des Ernteguts vom Feld bis zu den Lagerstätten über das Straßennetz. Die Planung der benötigten Ressourcen und der raumzeitlichen Kooperationen von Transportfahrzeugen, so wie die Anbindung an das beschriebene Planungssystem stellt den nächsten Schritt auf dem Weg der logistischen Planung von Ernteprozessen dar.

**Flexible Anzahl an Fahrzeugen:** Das Planungssystem wurde für einen Ernteprozess mit einem Ernte- und einem Überladefahrzeug entwickelt. In realen Ernteszenarien werden aber bei großen Feldern häufig mehrere dieser Fahrzeuge eingesetzt. Die Planung für eine flexible Anzahl an Feldfahrzeugen stellt daher eine sinnvolle und notwendige Erweiterung des Planungssystem für die Praxistauglichkeit dar. Dafür müssten die eingesetzten Verfahren

zur Routenplanung angepasst und erweitert werden. Zudem kommt es in der Praxis vor, dass sich die Zusammensetzung einer Fahrzeugflotte, die ein Feld bearbeitet, während der Prozessausführung verändert. Die Planung mit dynamischen Ressourcen stellt also ebenfalls eine relevante Problemstellung dar.

**Transfer der Planungsverfahren:** Die für den Anwendungsfall der Infield-Transportlogistik entwickelte raumzeitliche Planung könnte funktional in andere Anwendungen kooperativer mobiler autonomer Roboter transferiert werden. Denkbar sind beispielsweise Szenarien kooperativer Explorationen oder eine verallgemeinerte raumzeitlich Planung von Treffpunkten für kooperative Handlungen bestimmter Maschinengruppen.

# Literaturverzeichnis


[1] Acar, E. U.; Choset, H.; Zhang, Y.; Schervish, M.: Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods. In: *The International Journal of Robotics Research* 22 (2003), Nr. 7-8, S. 441–466

[2] Ackland, B. D.; Weste, N. H.: The Edge Flag Algorithm: A Fill Method for Raster Scan Displays. In: *IEEE Trans. Comput.* 30 (1981), Januar, Nr. 1, S. 41–48

[3] Ali, O.; Verlinden, B.; Van Oudheusden, D.: Infield logistics planning for crop-harvesting operations. In: *Engineering Optimization* 41 (2009), Nr. 2, S. 183–197

[4] Arkin, E. M.; Bender, M. A.; Demaine, E. D.; Fekete, S. P.; Mitchell, J. S.; Sethia, S.: Optimal covering tours with turn costs. In: *SIAM Journal on Computing* 35 (2005), Nr. 3, S. 531–566

[5] Arkin, R. C.: Cooperation without communication: Multiagent schema-based robot navigation. In: *Journal of Robotic Systems* 9 (1992), Nr. 3, S. 351–364

[6] Arleo, A.; R Millan, J. del; Floreano, D.: Efficient learning of variable-resolution cognitive maps for autonomous indoor navigation. In: *Robotics and Automation, IEEE Transactions on* 15 (1999), Nr. 6, S. 990–1000

[7] Asano, T.; Asano, T.; Guibas, L.; Hershberger, J.; Imai, H.: Visibility-polygon search and euclidean shortest paths. In: *Foundations of Computer Science, 1985., 26th Annual Symposium on*, 1985

[8] Bakhtiari, A. A.; Navid, H.; Mehri, J.; Bochtis, D. D.: Optimal route planning of agricultural field operations using ant colony optimization. In: *Agricultural Engineering International: CIGR Journal* 13 (2012), Nr. 4

[9] Barraquand, J.; Kavraki, L. E.; Latombe, J.-C.; Li, T.-Y.; Motwani, R.; Raghavan, P.: A Random Sampling Scheme for Path Planning. In: *INTERNATIONAL JOURNAL OF ROBOTICS RESEARCH* 16 (1996), S. 759–774

[10] Beckers, R.; Holland, O.; Deneubourg, J.-L.: From local actions to global tasks: Stigmergy and collective robotics. In: *Artificial life IV* Bd. 181, 1994

[11] Berg, M. de; Kreveld, M. van; Overmars, M.; Schwarzkopf, O.: *Computational Geometry: Algorithms and Applications*. 2. Springer-Verlag, 2000

[12] Blum, C.; Li, X.: *Swarm intelligence in optimization*. Springer, 2008

[13] Bochtis, D.; Sørensen, C.: The vehicle routing problem in field logistics part I. In: *Biosystems Engineering* 104 (2009), Nr. 4, S. 447–457

[14] Bochtis, D.; Sørensen, C.: The vehicle routing problem in field logistics: Part {II}. In: *Biosystems Engineering* 105 (2010), Nr. 2, S. 180 – 188

[15] Bochtis, D.; Vougioukas, S.; Ampatzidis, Y.; Tsatsarelis, C.: Field operations planning for agricultural vehicles: A hierarchical modeling framework. In: *Agricultural Engineering International: CIGR Journal* (2007)

[16] Braitenberg, V.: *Vehicles: Experiments in synthetic psychology*. MIT press, 1986

[17] Brambilla, M.; Ferrante, E.; Birattari, M.; Dorigo, M.: Swarm robotics: a review from the swarm engineering perspective. In: *Swarm Intelligence* 7 (2013), Nr. 1, S. 1–41

[18] Bryson, A. E.: *Applied optimal control: optimization, estimation and control*. CRC Press, 1975

[19] Canny, J.: *The Complexity of Robot Motion Planning*. 1988

[20] Canny, J.; Donald, B.: Simplified voronoi diagrams. In: *Discrete & Computational Geometry* 3 (1988), Nr. 1, S. 219–236

[21] Cao, Y. U.; Fukunaga, A. S.; Kahng, A.: Cooperative mobile robotics: Antecedents and directions. In: *Autonomous robots* 4 (1997), Nr. 1, S. 7–27

[22] Chazelle, B.: Approximation and Decomposition of Shapes. In: Schwartz, J. T. (Hrsg.); Yap, C. K. (Hrsg.): *Algorithmic and Geometric Aspects of Robotics*. Hillsdale, NJ : Lawrence Erlbaum Associates, 1987, S. 145–185

[23] Choset, H.: Coverage for robotics–A survey of recent results. In: *Annals of mathematics and artificial intelligence* 31 (2001), Nr. 1-4, S. 113–126

[24] Choset, H. M.: *Principles of robot motion: theory, algorithms, and implementations*. MIT press, 2005

[25] CLAAS GmbH: *Pressebilder*. Webseite, 2014. – Online verfügbar unter `http://www.claas-gruppe.com/presse/medien/pressebilder`; Stand 20.03.2014

[26] Dijkstra, E. W.: A note on two problems in connexion with graphs. In: *Numerische Mathematik* 1 (1959), Nr. 1, S. 269–271

[27] Domschke, W.; Drexl, A.: *Einführung in operations research*. Bd. 5. Springer, 2005

[28] Dorigo, M.; Birattari, M.; Stutzle, T.: Ant colony optimization. In: *Computational Intelligence Magazine, IEEE* 1 (2006), Nr. 4, S. 28–39

[29] Drogoul, A.; Ferber, J.: From tom thumb to the dockers: Some experiments with foraging robots. In: *From Animals to Animats II* (1993), S. 451–459

[30] Dubins, L. E.: On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. In: *American Journal of mathematics* (1957), S. 497–516

[31] Engelbrecht, A. P.: *Computational intelligence: an introduction*. John Wiley & Sons, 2007

[32] Feigenbaum, E. A.; Barr, A.; Cohen, P. R.: *The handbook of artificial intelligence*. Addison-Wesley New York, 1989

[33] Fiorelli, E. A.: *Cooperative vehicle control, feature tracking and ocean sampling*. Princeton University, 2005

[34] Fraichard, T.; Scheuer, A.: From Reeds and Shepp's to continuous-curvature paths. In: *Robotics, IEEE Transactions on* 20 (2004), Nr. 6, S. 1025–1035

[35] Frangioni, A.; Manca, A.: A computational study of cost reoptimization for min-cost flow problems. In: *INFORMS Journal on Computing* 18 (2006), Nr. 1, S. 61–70

[36] Frese, C.: *Planung kooperativer Fahrmanöver für kognitive Automobile*. Bd. 10. KIT Scientific Publishing, 2011

[37] Goss, S.; Deneubourg, J.-L.: Harvesting by a group of robots. In: *Proceedings of the First European Conference on Artificial Life*, 1992

[38] Grossman, D. D.: Traffic control of multiple robot vehicles. In: *Robotics and Automation, IEEE Journal of* 4 (1988), Nr. 5, S. 491–497

[39] Hameed, I. A.; Bochtis, D.; Sørensen, C. A.: An Optimized Field Coverage Planning Approach for Navigation of Agricultural Robots in Fields Involving Obstacle Areas. In: *International Journal of Advanced Robotic Systems* 10 (2013)

[40] Hart, P.; Nilsson, N.; Raphael, B.: A Formal Basis for the Heuristic Determination of Minimum Cost Paths. In: *Systems Science and Cybernetics, IEEE Transactions on* 4 (1968), July, Nr. 2, S. 100–107

[41] Hartanto, R.; Scheuren, S.: *Planung in marion.* Vortrag auf der ersten Meilensteinveranstaltung des Projekts marion, 2013

[42] Hertzberg, J.; Lingemann, K.; Nüchter, A.: *Mobile Roboter: Eine Einführung aus Sicht der Informatik.* Springer-Verlag GmbH, 2012

[43] Hoffman, W.; Martin, K.: The CMake Build Manager. In: *Dr. Dobb's Journal* 28 (2003), 01, Nr. 1, S. 40–47

[44] Hsu, D.; Latombe, J.-C.; Motwani, R.: Path planning in expansive configuration spaces. In: *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on* Bd. 3, 1997

[45] Jensen, M. A. F.; Bochtis, D.; Sørensen, C. G.; Blas, M. R.; Lykkegaard, K. L.: In-field and inter-field path planning for agricultural transport units. In: *Computers & Industrial Engineering* 63 (2012), Nr. 4, S. 1054–1061

[46] Kanayama, Y.; Hartman, B. I.: *Smooth local path planning for autonomous vehicles.* Springer, 1990

[47] Karaman, S.; Frazzoli, E.: Sampling-based Algorithms for Optimal Motion Planning. In: *CoRR* abs/1105.1186 (2011)

[48] Kavraki, L. E.; Kolountzakis, M. N.; Latombe, J.-C.: Analysis of probabilistic roadmaps for path planning. In: *Robotics and Automation, IEEE Transactions on* 14 (1998), Nr. 1, S. 166–171

[49] Kavraki, L. E.; LaValle, S. M.: Motion Planning. In: Siciliano, B. (Hrsg.); Khatib, O. (Hrsg.): *Springer Handbook of Robotics.* Springer Berlin Heidelberg, 2008, S. 109–131

[50] Kavraki, L. E.; Svestka, P.; Latombe, J.-C.; Overmars, M. H.: Probabilistic roadmaps for path planning in high-dimensional configuration spaces. In: *Robotics and Automation, IEEE Transactions on* 12 (1996), Nr. 4, S. 566–580

[51] Kelly, A.; Nagy, B.: Reactive nonholonomic trajectory generation via parametric optimal control. In: *The International Journal of Robotics Research* 22 (2003), Nr. 7-8, S. 583–601

[52] Komoriya, K.; Tanie, K.: Trajectory design and control of a wheel-type mobile robot using B-spline curve. In: *Intelligent Robots and Systems' 89. The Autonomous Mobile*

*Robots and Its Applications. IROS'89. Proceedings., IEEE/RSJ International Workshop on* IEEE, 1989

[53] Land, A. H.; Doig, A. G.: An Automatic Method of Solving Discrete Programming Problems. In: *Econometrica* 28 (1960), Nr. 3, S. pp. 497–520

[54] Latombe, J.-C.: *Robot Motion Planning.* Norwell, MA, USA : Kluwer Academic Publishers, 1991

[55] LaValle, S. M.: *Planning Algorithms.* Cambridge, U.K. : Cambridge University Press, 2006

[56] Lavalle, S. M.; Kuffner, J. J.; Jr.: Rapidly-Exploring Random Trees: Progress and Prospects. In: *Algorithmic and Computational Robotics: New Directions*, 2000

[57] Likhachev, M.: *Search-based Planning with Motion Primitives.* Presentation at ROS Cotesys School at TUM, 2010. – Available online at `http://www.ros.org/wiki/Events/CoTeSys-ROS-School?action=AttachFile&do=get&target=robschooltutorial_oct10.pdf`

[58] Likhachev, M.: *Search-based planning library (SBPL) Paket für ROS.* Website, 2014. – Available online at `http://wiki.ros.org/sbpl`; Stand 20.03.2014

[59] Likhachev, M.; Ferguson, D.: Planning long dynamically feasible maneuvers for autonomous vehicles. In: *The International Journal of Robotics Research* 28 (2009), Nr. 8, S. 933–945

[60] Liscano, R.; Green, D.: Design and implementation of a trajectory generator for an indoor mobile robot. In: *Intelligent Robots and Systems' 89. The Autonomous Mobile Robots and Its Applications. IROS'89. Proceedings., IEEE/RSJ International Workshop on* IEEE, 1989

[61] Lumelsky, V. J.; Stepanov, A. A.: Path-planning strategies for a point mobile automaton moving amidst unknown obstacles of arbitrary shape. In: *Autonomous robot vehicles.* Springer, 1990, S. 363–390

[62] Martins, E. Q. V.: On a multicriteria shortest path problem. In: *European Journal of Operational Research* 16 (1984), Nr. 2, S. 236–245

[63] Mataric, M. J.: Integration of representation into goal-driven behavior-based robots. In: *Robotics and Automation, IEEE Transactions on* 8 (1992), Nr. 3, S. 304–312

[64] Mataric, M. J.; Nilsson, M.; Simsarin, K.: Cooperative multi-robot box-pushing. In: *Intelligent Robots and Systems 95.'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on* Bd. 3 IEEE, 1995

[65] Meyer, J.-A.; Filliat, D.: Map-based navigation in mobile robots:: II. a review of map-learning and path-planning strategies. In: *Cognitive Systems Research* 4 (2003), Nr. 4, S. 283–317

[66] Michael, N.; Fink, J.; Kumar, V.: Cooperative manipulation and transportation with aerial robots. In: *Autonomous Robots* 30 (2011), Nr. 1, S. 73–86

[67] Misra, P.; Enge, P.: *Global Positioning System: Signals, Measurements and Performance Second Edition*. Massachusetts: Ganga-Jamuna Press, 2006

[68] Murphy, R. R.: *Introduction to AI Robotics*. 1st. Cambridge, MA, USA : MIT Press, 2000

[69] Nelson, W.: Continuous-curvature paths for autonomous vehicles. In: *Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on* IEEE, 1989

[70] Ogren, P.; Leonard, N. E.: Obstacle avoidance in formation. In: *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on* Bd. 2 IEEE, 2003

[71] Oksanen, T.; Visala, A.: Coverage path planning algorithms for agricultural field machines. In: *Journal of Field Robotics* 26 (2009), Nr. 8, S. 651–668

[72] OMG: Data-Distribution Service for Real-Time Systems (DDS).v1.2 / Object Management Group (OMG). 2007. – Forschungsbericht

[73] O'rourke, J.: *Art gallery theorems and algorithms*. Bd. 1092. Oxford University Press Oxford, 1987

[74] Parker, L. E.: Path planning and motion coordination in multiple mobile robot teams. In: *Encyclopedia of Complexity and System Science* (2009)

[75] Pivtoraiko, M.; Knepper, R. A.; Kelly, A.: Optimal, smooth, nonholonomic mobile robot motion planning in state lattices. In: *Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-07-15* (2007)

[76] Poli, R.; Kennedy, J.; Blackwell, T.: Particle swarm optimization. In: *Swarm intelligence* 1 (2007), Nr. 1, S. 33–57

[77] PrismTech: *OpenspliceDDS Produktwebseite*. Webseite, 2014. – Online verfügbar unter `http://www.prismtech.com/opensplice/`; Stand 20.03.2014.

[78] Qt Project: *Projektwebseite*. Webseite, 2014. – Online verfügbar unter `http://qt-project.org/`; Stand 20.03.2014.

[79] Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A. Y.: ROS: an open-source Robot Operating System. In: *ICRA workshop on open source software* Bd. 3, 2009

[80] Reeds, J.; Shepp, L. u. a.: Optimal paths for a car that goes both forwards and backwards. In: *Pacific Journal of Mathematics* 145 (1990), Nr. 2, S. 367–393

[81] Reinecke, M.; Scheuren, S.: *Anwendungsfeld Landwirtschaft.* Vortrag auf dem Abschlussmeilenstein des Projekts marion, 2013. – online verfügbar unter `http://www.projekt-marion.de/`

[82] Ruckelshausen, A.; Biber, P.; Dorna, M.; Gremmes, H.; Klose, R.; Linz, A.; Rahe, F.; Resch, R.; Thiel, M.; Trautz, D. u. a.: BoniRob–an autonomous field robot platform for individual plant phenotyping. In: *Precision agriculture* 9 (2009), S. 841

[83] Russell, S. J.; Norvig, P.: *Artificial Intelligence: A Modern Approach.* 2. Pearson Education, 2003

[84] Sasaki, J.; Ota, J.; Yoshida, E.; Kurabayashi, D.; Arai, T.: Cooperating grasping of a large object by multiple mobile robots. In: *Robotics and Automation, 1995. Proceedings., 1995 IEEE International Conference on* Bd. 1 IEEE, 1995

[85] Scheuren, S.; Hertzberg, J.; Stiene, S.; Hartanto, R.: Infield Path Planning for Autonomous Unloading Vehicles. In: Clasen, M. (Hrsg.): *Referate der 33. GIL-Jahrestagung in Potsdam 2013 - Massendatenmanagement in der Agrar- und Ernährungswirtschaft. Gesellschaft für Informatik in der Land-, Forst- und Ernährungswirtschaft (GIL-2013), February 20, Potsdam, Germany*, Köller Druck+Verlag GmbH, 2 2013

[86] Scheuren, S.; Stiene, S.; Hartanto, R.; Hertzberg, J.: Approximating Reference Trajectories for Autonomous Vehicles using Motion Primitives. In: *Autonome Mobile Systeme (AMS-12), September 26, Stuttgart, Germany*, Springer, 9 2012

[87] Scheuren, S.; Stiene, S.; Hartanto, R.; Hertzberg, J.; Reinecke, M.: The Problem of Spatio-temporally Constrained Motion Planning for Cooperative Vehicles. In: *Proc. 26. Workshop Planen, Scheduling und Konfigurieren, Entwerfen (PuK2011)*, 2011

[88] Scheuren, S.; Stiene, S.; Hartanto, R.; Hertzberg, J.; Reinecke, M.: Spatio-temporally Constrained Planning for Cooperative Vehicles in a Harvesting Scenario. In: *KI - Künstliche Intelligenz, German Journal on Artificial Intelligence* 27 (2013), Nr. 4, S. 341–346

[89] Sen, S.; Sekaran, M.; Hale, J. u. a.: Learning to coordinate without sharing information. In: *AAAI*, 1994

[90] Spekken, M.; Bruin, S. de: Optimized routing on agricultural fields by minimizing maneuvering and servicing time. In: *Precision Agriculture* 14 (2013), Nr. 2, S. 224–244

[91] Sreenath, K.; Kumar, V.: Dynamics, Control and Planning for Cooperative Manipulation of Payloads Suspended by Cables from Multiple Quadrotor Robots. In: *rn* 1 (2013), Nr. r2, S. r3

[92] Steels, L.: Cooperation between distributed agents through self-organisation. In: *Decentralized A.I.: proceedings of the First European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Cambridge, England, August 16–18, 1989* pub-ELS-NH, 1990

[93] STILL GmbH: *Bildarchiv*. Webseite, 2014

[94] Stone, P.; Veloso, M.: Multiagent systems: A survey from a machine learning perspective. In: *Autonomous Robots* 8 (2000), Nr. 3, S. 345–383

[95] Sugawara, K.; Kazama, T.; Watanabe, T.: Foraging behavior of interacting robots with virtual pheromone. In: *Intelligent Robots and Systems, 2004.(IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on* Bd. 3 IEEE, 2004

[96] Švestka, P.; Overmars, M. H.: Coordinated path planning for multiple robots. In: *Robotics and autonomous systems* 23 (1998), Nr. 3, S. 125–152

[97] Takahashi, A.; Hongo, T.; Ninomiya, Y.; Sugimoto, G.: Local path planning and motion control for AGV in positioning. In: *Intelligent Robots and Systems' 89. The Autonomous Mobile Robots and Its Applications. IROS'89. Proceedings., IEEE/RSJ International Workshop on* IEEE, 1989

[98] Thrun, S.; Leonard, J. J.: Simultaneous Localization and Mapping. In: Siciliano, B. (Hrsg.); Khatib, O. (Hrsg.): *Springer Handbook of Robotics*. Springer Berlin Heidelberg, 2008, S. 871–889

[99] Tung, B.; Kleinrock, L.: Distributed control methods. In: *High Performance Distributed Computing, 1993., Proceedings the 2nd International Symposium on* IEEE, 1993

[100] Tzafestas, S. G.: *Introduction to Mobile Robot Control*. Elsevier, 2013

[101] Ullrich, A.; Hertzberg, J.; Stiene, S.: ROS-Based Path Planning and Machine Control for an Autonomous Sugar Beet Harvester. In: *Proceedings of International Conference on Machine Control & Guidance. International Conference on Machine Control & Guidance (MCG-2014), March 19-20, Braunschweig, Germany*, 2014

[102] Vlassis, N.: A concise introduction to multiagent systems and distributed artificial intelligence. In: *Synthesis Lectures on Artificial Intelligence and Machine Learning* 1 (2007), Nr. 1, S. 1–71

[103] Weicker, K.: *Evolutionäre Algorithmen.* Springer, 2007

[104] Willow Garage: *Produktwebseite PR2 Roboter.* Webseite, 2014. – Online verfügbar unter `http://www.willowgarage.com/pages/pr2/overview`; Stand 20.03.2014

[105] Wolpert, D. H.; Macready, W. G.: No free lunch theorems for optimization. In: *Evolutionary Computation, IEEE Transactions on* 1 (1997), Nr. 1, S. 67–82

[106] Zelinsky, A.; Jarvis, R. A.; Byrne, J.; Yuta, S.: Planning paths of complete coverage of an unstructured environment by a mobile robot. In: *Proceedings of international conference on advanced robotics* Bd. 13, 1993

# Erklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Osnabrück, März 2014