Projekt
Autonome Mobile Systeme

Dokumentation

Wintersemester 2010/11

Stefan Dieckmann, Daniel Kiertscher, Daniel Kiewitz

Anfangsphase

Woche 1-3

In der ersten Woche gab es zunächst eine erste Einführung für das Labor, das Aksen-Board, eine Vorstellung von verschieden Projekten der vergangenen Jahrgänge sowie natürlich unsere Aufgabenstellung für das Projekt „Autonome Mobile Systeme“ im Wintersemester 2010/11.

Zielstellung dabei ist es einen autonomen mobilen Roboter sowohl mechanisch als auch softwareseitig so zu konstruieren, dass er einen Parkuhr durchqueren und am Ziel angelangt 3 symbolische Methan-Moleküle spalten kann. Dies geschieht indem sie von ihrem Kohlenstoff-Atom getrennt werden.
Um die Sache spannender zu gestalten treten dabei immer 2 Roboter gegeneinander in einem Rennen an. Jedes der 3 Methan-Moleküle kann nur von einem Kontrahenten gespalten werden, wofür es den Hauptanteil an Punkten (jeweils 10) gibt. Weitere Punkte gibt es fürs Losfahren bei Lichtsignal (1 Punkt) sowie jeweils 3 Punkte für das Erreichen eines neuen Checkpoints auf dem Spielfeld.

Nebenbedingungen der Aufgabenstellung sind:

- Es dürfen keine Bauteile verloren oder abgelegt werden, wenn sich diese nach Spiel ende noch auf dem Feld befinden
- Die Zeitdauer beträgt 120 Sekunden, es darf weder vorher gestartet, noch nach Zeitablauf weiteragiert werden
- Keine Informationsübermittlung während des Spiels von Teammitgliedern an den Roboter
- Der Roboter darf weder Menschen noch Roboter angreifen

Erste Überlegungen an die Anforderungen des Roboters ergaben:

- Der Roboter muss geradeaus fahren können
- Eine Drehung muss möglich sein, um Kurven nehmen zu können
- Es wird eine Stoßvorrichtung benötigt um die Methan-Moleküle zu spalten
- Der Startzeitpunkt muss erfasst werden können
- Hindernisse wie Wände werden erkannt
- Der markierte Weg auf dem Spielfeld muss verfolgt werden
- Gegen Ende des Spielfeldes ist der Weg nicht mehr markiert, auch hier muss der Weg gefunden werden
- Eine stabile Bauweise ist nötig um keine Bauteile zu verlieren und um der allgemeinen mechanischen Belastung standzuhalten

Mögliche Lösungen für diese Aufgaben wurden erdacht:

- Die Wegmarkierung kann als Orientierung für Geradeausfahrt genutzt werden
- In Anlehnung an bestehende Roboter wurde ein Antrieb mit 2 Motoren, sowie 2 Rädern ausgewählt, dadurch ergibt sich die Möglichkeit sich auf der Stelle drehen zu können
- Mittels Helligkeitssensoren und LEDs könnten Hindernisse erfasst werden
- Der markierte Weg könnte mit den gleichen Sensortypen erfasst werden
- Eine ausfahrbare Stoßvorrichtung sollte von Vorteil sein, um während der Fahrt nicht sich selbst zu blockieren und trotzdem am Ende effektiv die Moleküle spalten zu können
- Ein Helligkeitssensor am Boden kann die Startlampe erfassen

So begann der **Bau eines ersten Prototyps** des Roboters, noch ohne Sensoren, wobei als Vorbild für die Grundkonstruktion ein bereits bestehender Roboter diente.

Nach ersten **Testläufen** ergaben sich dann **weitere Erkenntnisse**:

- Eine exakte Geradausfahrt ist aufgrund von Fertigungstoleranzen bei den Motoren und Rädern erst nach einigen Proben unterschiedlicher Exemplare zu erreichen
- Das Getriebe ist auf relativ hohe Geschwindigkeiten ausgelegt, eine kontrollierte, kraftvolle, langsame Fahrt fast nicht möglich, die Getriebeuntersetzung wurde also überarbeitet

Nach dem Lösen der Antriebsprobleme widmeten wir uns der **Sensorik**:

- Die **Streckenmarkierung** muss während der Fahrt erkannt werden, um den aktuellen Standpunkt zu bestimmen, sowie feststellen zu können ob eine gewünschte Fahrrichtung eingehalten wird
- Dazu wurde ein System mit 9 Helligkeits-Sensoren auf einer horizontalen Achse erdacht

Jetzt wird es ernst! – Kernphase

Die 9 Sensoren wurden auf einer modifizierten Kunststoffleiste mit Heißkleber fixiert.

- ✓ Es wurde nach einigen Tests und Überlegungen festgestellt dass mit lediglich 6 Sensoren die gesamte Sensorik erfolgen kann [Foto2]
- ✓ allein 3 davon für die Geradeausfahrt: Es wird mit **3 Sensoren in der Mitte der Fahrzeugfront** erkannt wann der Kurs von der Ideallinie abweicht und entsprechend, mit einer Abbremsung des richtigen Motors, der Kurs korrigiert. Dieses System erweist sich später als äußerst robust und zuverlässig.
- ✓ **2 weitere Sensoren erkennen** an einer „Kreuzung" auf dem Spielfeld die Markierungslinien.
- ✓ Diese werden gezählt und an den richtigen „**Kreuzungen**" wird dann, mit den Motoren, der Roboter auf der Stelle gedreht.
- ✓ Einer der 3 mittleren Sensoren erkennt in der Drehung die neue zu befolgende Linie. Daraufhin folgt der Roboter dieser neuen Markierung.
- ✓ Der **6. Sensor** dient lediglich der **Erkennung der Startlampe** und ist in der Fahrzeugmitte nach unten „sehend" montiert.

Problematisch dabei waren insbesondere:

- ☞ Der **Algorithmus für die Geradeausfahrt** funktionierte am besten indem bei Abweichung ein Motor komplett abgeschaltet wird, dies erfolgt in jedem Programmdurchlauf, daraus folgt eine extrem hohe Schaltfrequenz. Nur eine Reduzierung einer Motorspannung brachte, zur Verwunderung der Tester, nicht die erwünschten Ergebnisse.
- ☞ Ähnlich verhielt es sich bei der **Drehung auf der Kreuzung**. Es musste die Position der äußeren Sensoren ermittelt werden, in welcher der Roboter bei Erkennung einer Linie und folgender Drehung auch relativ gerade auf die andere Linie der Kreuzung drehen kann.
- ☞ Sind die Sensoren zu weit vorne dreht der Roboter zu früh, er würde mit den mittleren Sensoren die neue Linie zu früh in der Drehung erkennen, z.B. bereits nach nur 60° statt 90° Drehung. Andersrum verhält es sich, wenn die Sensoren zu weit hinten angebracht sind.
- ☞ Als optimal hat sich eine Position kurz vor den Antriebsrädern erwiesen.[Foto2]

Nach der Meisterung der ersten Kurven wurde der Ablauf bis zum Ende des markierten Fahrweges implementiert.

- ✓ Bereits **im 1. Durchlauf** erreichte der Roboter die erwartet Position
- ✓ Eine Markierung gilt als überquert wenn ein Sensor die Linie und anschließend das Verlassen der Linie erkennt.
- ✓ Die Linienzählung kann mit 2 Sensoren durchgeführt werden, dazu wurde ein **Sperrverfahren** implementiert, sodass jeweils nur ein Sensor die Zählung einer Linie ausführen kann
- ✓ Auch eine **Zeitsperre** die sicherstellt, dass nicht in wenigen Millisekunden fälschlicherweise mehrere Linien gezählt werden können wurde implementiert.
- ✓ Die **Verschmutzung des Spielfeldes** machte Probleme bei der korrekten Erkennung der Markierung, da der Roboter oftmals Schmutz mit einer Linie verwechselte. Der Grenzwert für die schwarze Farbe wurde daraufhin korrigiert.

Nach einigen Durchläufen und Korrekturen am Code meisterte der Roboter den Kurs bis hierhin mit sehr hoher Zuverlässigkeit.

- ☞ Neue Probleme ergaben sich mit der „freien" Drehung bei Wegpunkt E, es stehen nun **keine Markierungen** zum Befolgen der Linie mehr bereit. [Abbildung1]
- ☞ Zur Lösung wurde die Zeit gemessen die der Roboter bei der vorherigen Kurve in die gleiche Richtung benötigte. Diese Zeit wird dann für die „freie" Drehung ebenfalls verwendet.
- ☞ **Zeitverzögerungen im Programmablauf** und die ebenfalls **nicht konstante Drehgeschwindigkeit der Motoren** in beiden Drehungen führen trotzdem zu einer nicht einwandfreien 90°-Drehung
- ☞ Es wird versucht den **Kurs** zu **korrigieren** wenn die nächste Kreuzungsmarkierung überquert wird.
- ☞ Dazu wird überprüft ob ein mittlerer Sensor die Linie ebenfalls erkennt, wenn dies der Fall ist wird nach außen korrigiert, andernfalls nach innen.
- ☞ Wichtig für die Überlegung dieser Regelung ist die Feststellung, dass der Roboter **niemals** genau **geradeaus** fährt und deshalb immer korrigiert werden muss.

- ☞ Denkbar wäre eine **noch feinere Unterteilung**, wenn nur von 1 oder 2 der mittleren Sensoren die Markierungslinie erkannt wird.
- ☞ Zudem wurde stets eine **Neigung des Roboters nach rechts abzuweichen** festgestellt. Eine Korrektur nach rechts muss daher vom Programm ausgeschlossen werden, da dies die Situation stets nur Verschlimmern würde.

Der Roboter erreicht mit dieser Methode recht zuverlässig die letzte Gerade.

- ☞ Jedoch treten manchmal nicht reproduzierbare starke Abweichungen während der Fahrt ohne Bodenmarkierung auf.

Auf der Zielgrade gilt es nun die Moleküle, durch Herunterstoßen der Kugeln, zu trennen.

- ✓ Dazu wurde auf Grundlage eines Servomotors eine Stoßvorrichtung konstruiert, die bis zum Erreichen der Zielgrade nach oben zeigt und erst am Ende nach links oder rechts ausklappt.
- ✓ Dadurch ist es möglich eine lange, und damit effektive, Stoßkonstruktion zu nutzen, die aber während der Fahrt nicht das gesamte Fahrzeug an Hindernissen hängen bleiben lässt.
- ☞ Problematisch dabei war das hohe Gewicht des Stoßarmes, der für Servomotor zu schwer war. Es gelang nicht zuverlässig den Arm am Start nach oben zu drehen und zu halten.
- ☞ Es musste eine stabile und trotzdem sehr leichte Gesamtkonstruktion gefunden werden, um der geringen Kraft des Servomotors Rechnung tragen zu können.
- ✓ Dies gelang schließlich mit einem hohlen Kunststoffrohr. Jedoch muss trotzdem für die jeweilige Startposition das Kunststoffrohr umgesteckt werden, damit am Ende eine entspreche Höhe zum Herunterstoßen der Kugeln erreicht werden kann.

Endphase – Testen, Testen & Testen

Das Programm steuert den Roboter nun recht zuverlässig bis zur Endgerade und lässt ihn dann die Kugeln herunterstoßen

- ☞ Jedoch traten oft seltsame Abweichungen auf für die es nun Ursachen zu finden galt.
- ☞ Eine Ursache konnte, mit Hilfe eines Voltmeters, auf die **unterschiedliche Leistungsfähigkeit** der beiden **Akku-Pakete** zurückgeführt werden.
- ✓ Als Workaround wurden einige Akkus zwischen den Paketen getauscht. So wurde eine annähernd gleiche Leistungsfähigkeit erreicht. Konstantere Ergebnisse bei den Testläufen stellten sich ein.

Vorstellung der Roboter-Hardware

Foto 1: der Roboter in Totale

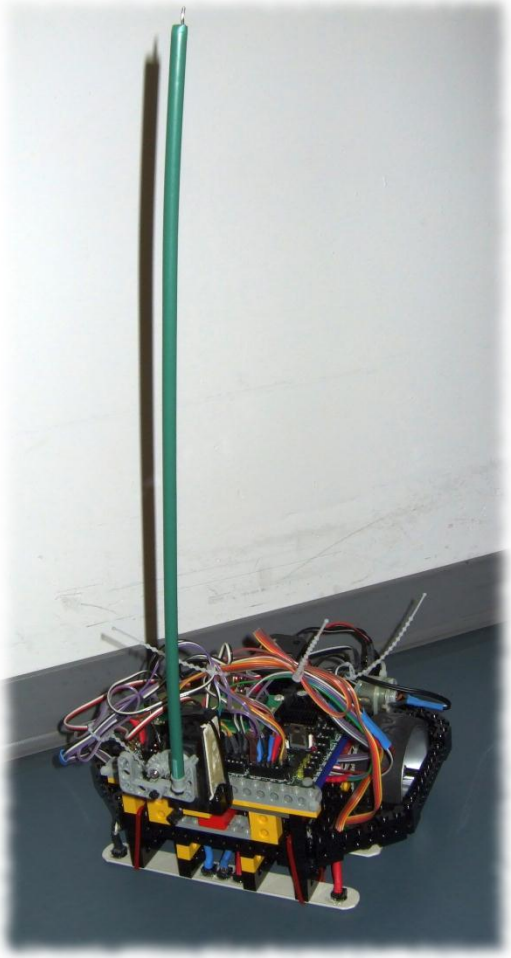
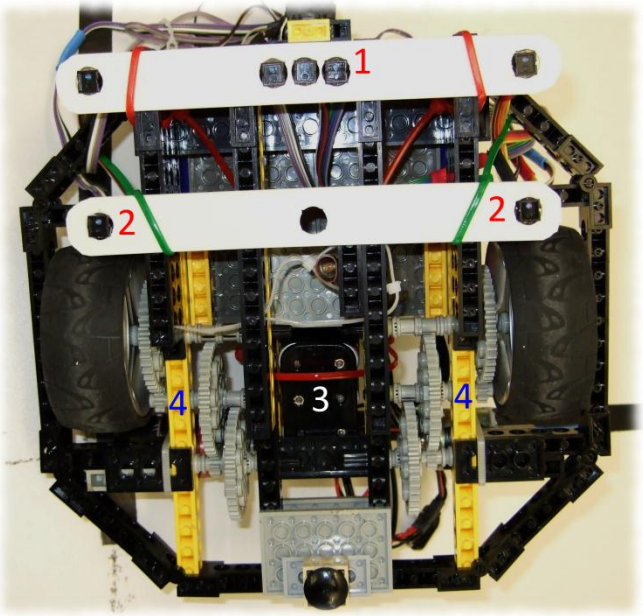


Foto 2: Roboter von unten



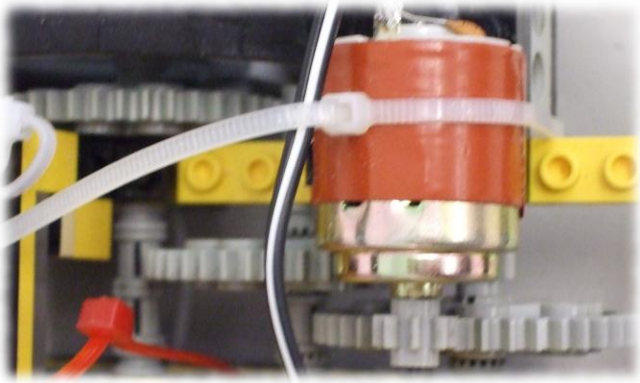


Foto 3: Antrieb im Detail

Auf den Fotos erkennt man den Roboter als Ganzes mit langem Stoßarm, sowie eine Sicht von unten auf die Sensoren für Geradeausfahrt(1), Kreuzungserkennung(2), die Batterieeinheit(3) sowie den Antrieb(4). Außerdem eine Detailansicht des Antriebs mit variabler Motorenbefestigung, die einen Zahnradwechsel zur Getriebeveränderung schnell möglich macht.

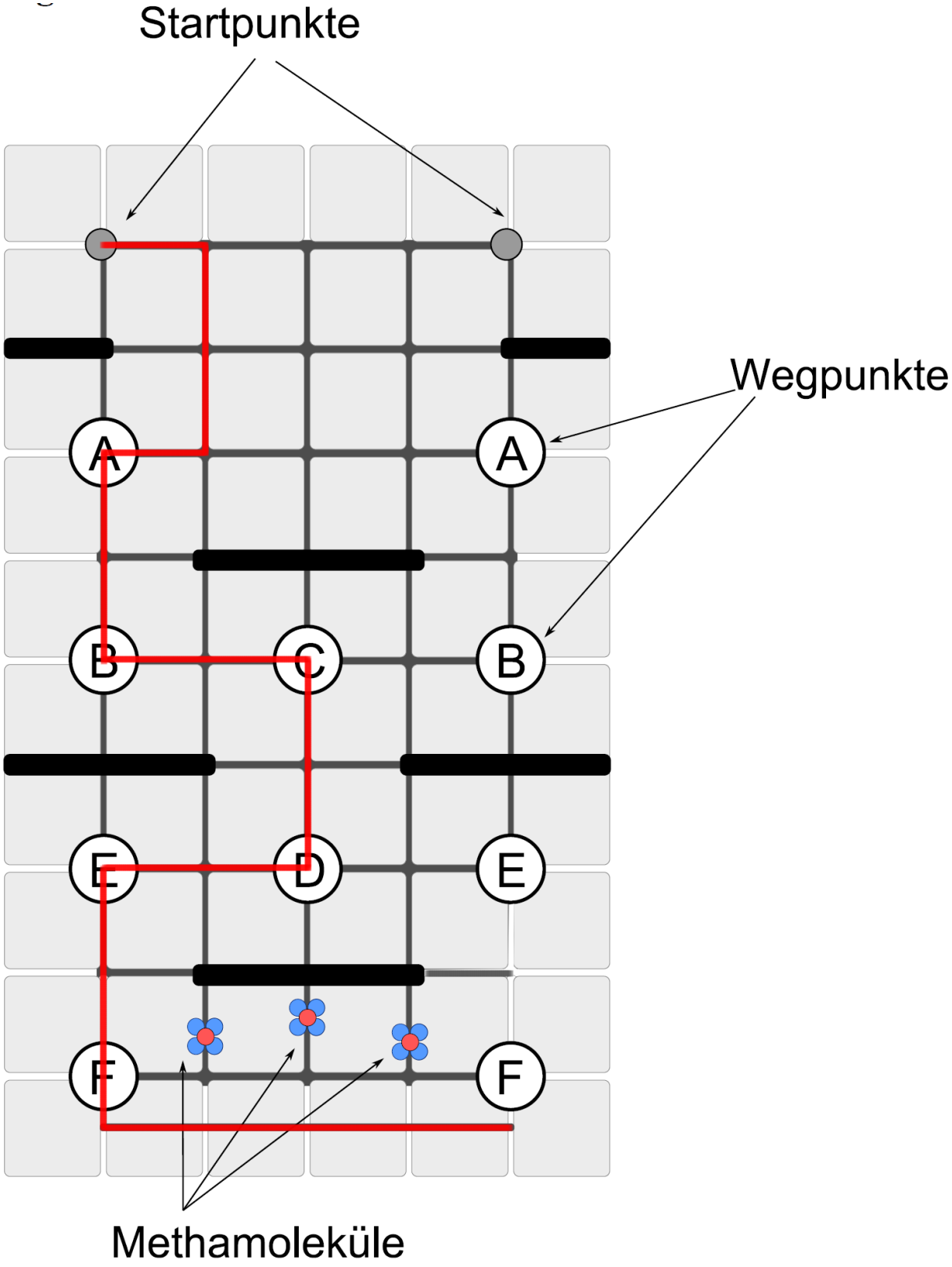


Abbildung 1:Spielfeld mit geplanter Fahrroute

Hardwareprobleme

- ☞ Das Getriebe muss ohne Schmierung auskommen, dadurch ergeben sich automatisch höhere Reibungsverluste
- ☞ Die Motorwellen laufen teilweise etwas unrund->Die Zahnräder müssen dies ausgleichen
- ☞ Der relativ lange und schwere Stoßarm macht es dem Servomotor nicht leicht diesen zielgenau zu bewegen
- ☞ Die Sensoren müssen möglichst tief angebracht werden um den Boden gut zu erkennen, da dieser jedoch auch teilweise uneben ist droht das Fahrzeug ständig aufzusetzen
- ☞ Lose Kabel dürfen nicht an bewegliche Teile geraten, bei der Vielzahl der Kabel ein wichtiger Aspekt

Einschätzung der Leistungsfähigkeit des Roboters

- ✓ Stabile Konstruktion: auch bei Zusammenstößen wurden keine Teile verloren oder gelockert, das Ergebnis von vielen Test und Anpassungen
- ✓ Sensoren können den Untergrund sehr gut erkennen, 1 defekter Sensor wurde ausgetauscht
- ✓ Der Servomotor kann die Stoßstange handeln
  - × Jedoch ist er nicht mehr so zuverlässig das hier von einer Dauerlösung gesprochen werden kann
- ✓ Die Softwarelösung war die erste die den Kurs im Rahmen des Semesterprojekt erfolgreich meisterte
  - × Die Geschwindigkeit auf dem Spielfeld ist dagegen ziemlich gering, die Konkurrenten haben Konstruktionen die den Kurs theoretisch deutlich schneller absolvieren können.
  - × Der Erfolg beruht zum Teil darauf, dass andere Ansätze nicht vollständig im Zeitrahmen des Projektes umgesetzt werden können.

Der Roboter im Wettbewerbsvergleich

- ✓ Als einziger Roboter absolviert er den Einzeldurchlauf fehlerfrei und sichert sich einen ersten Punktevorsprung
- ✓ Nach den Duellen führt der Roboter die Punkteliste an
  - ☞ obwohl der Roboter langsamer über das Feld schreitet als 2 Kontrahenten, in einer Runde klar verliert, weil er zu langsam ist, und in einer anderen Runde blockiert wird
  - ☞ Ausschlaggebend war die hohe Zuverlässigkeit, die den anderen Robotern fehlte

Finale!

Der Kontrahent sprintet unserem Roboter davon, verfehlt jedoch zwischen den Wegpunkten E und F seine Bahn und fährt gegen die Wand! Unserer Roboter schafft es ein Molekül zu spalten, bevor er in wenige Zentimeter weiter in den Kontrahenten fährt. Damit sichert si ch unser Roboter den Sieg im Wettbewerb.

2. Lösungsansatz (Name: Das Küken)

Zielstellung

Statt auf einer Linie zu pendeln, soll das Fahrzeug diagonal durch die Felder navigieren können und dabei weichere Kurven fahren und Wegstrecke sparen.

Lösungsansätze

An den Seitenlinien sowie jeder Querlinie und Ecke des Linienrasters den Kurs korrigieren.
Korrekturaktion während der Fahrt in einem Zeitfenster ausführen, oder das Fahrzeug an der Querlinie einseitig stoppen und ausrichten.

Voraussetzungen

Für alle auftretenden Sensorzustände muss eine gültige Regel zutreffen.
Dafür muss der Antrieb des Fahrzeugs so justiert werden können, dass er innerhalb eines Quadratfeldes ohne weitere Softwareunterstützung zuverlässig vom Eintritt zum Austritt eines Feldes fahren kann. Dies ist jedoch von der Geschwindigkeit und der Wirksamkeit der Korrekturaktionen abhängig.

Implementierung

Der Kurs auf der Bewegungsfläche wird durch ein Potentialfeld angegeben.
Die Sensorzustände innerhalb eines Feldes werden durch Funktionen repräs entiert.
In jedem Zustand kann ggf. eine Korrekturaktion ausgeführt werden.
Für den Aufruf des Folgezustandes werden Regeln für die aktuellen Sensorzustände formuliert.

Jeder Sensorzustand wird auf dem Display ausgegeben.
Auf diese Weise können die nötigen Korrekturaktionen im Verlauf der Erprobung des Fahrzeugs nach Bedarf formuliert und implementiert werden.
Stop ist die Standardaktion jedes Zustandes zu Beginn der Erprobungsphase .

Umsetzung

Der Zustandsraum wird durch die Repräsentation als Funktion und die konkrete Implementierung nach Bedarf handhabbarer.
Eine passende Unterstützung im Editor ermöglicht es, zu gleichnamigen Funktionen zu springen.
So ließen sich erst einmal die Zustandspaare für links und rechts im Editor im Zusammenhang bearbeiten. Verwendet wurde hierfür
Visual Studio 2008 mit dem AddOn rockscroll bzw.
Visual Studio 2010 mit dem AddOn AllMargins.
Für den Vim Editor und das cTags Plugin erhält man eine ähnliche Unterstützung.

aufgetretene Probleme

Die Justierung des Antriebs zum zuverlässigen überqueren eines Feldes bereitete Schwierigkeiten.
Die Begrenzung der Anzahl der zu implementierenden Korrekturaktionen für die Sensorzustände, hängt stark von der Anzahl und Anordnung der Sensoren am Fahrzeug ab.

Stand des Projekts

Das Projekt konnte nicht zum Abschluss gebracht werden,
da sich der Aufwand der Justierung des Antriebs und
die Implementierung der Korrekturaktionen und deren Erprobung
als umfangreicher herausgestellt hat als ursprünglich angenommen.