Bachelorarbeitarbeit

Johann Tschisow

Positionsbestimmung für mobile Systeme mit Hilfe eines Laserscanners, Mikrokontrollers und intelligenter Analogsensorik

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Johann Tschisow

Positionsbestimmung für mobile Systeme mit Hilfe eines Laserscanners, Mikrokontrollers und intelligenter Analogsensorik

Bachelorarbeitarbeit eingereicht im Rahmen der Bachelorarbeitprüfung
im Studiengang Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. rer. nat. Reinhard Baran
Zweitgutachter : Prof. Dr. rer. nat. Stephan Pareigis

Abgegeben am 23. April 2008

Johann Tschisow

Thema der Bachelorarbeit

Positionsbestimmung für mobile Systeme mit Hilfe eines Laserscanners, Mikrokontrollers und intelligenter Analogsensorik

Stichworte

Positionsbestimmung, Fitting-Bereich, geschätzte Position, Laserscanner, Sensorik, AT90CAN128

Kurzzusammenfassung

Sich in einer momentanen Umgebung zu orientieren und den genauen Standort zu bestimmen, ist eine der Grundvoraussetzungen für autonome mobile Systeme. Dabei hängen die Art und Weise, wie die Positionskoordinaten berechnet werden, von der Technik und Ausstattung des autonomen Roboters. In dieser Bachelorarbeit wird ein Modell vorgestellt, das die Ermittlung der exakten Positionskoordinaten des mobilen Fahrzeugs in einer eingegrenzten Umgebung aus den Messdaten des Laserscanners und der eingesetzten Sensoren ermöglicht.

Johann Tschisow

Title of the paper

Position determining for mobile systems with help of a laserscanner, microcontroller and intelegent analogue sensoric.

Keywords

Position determining, Fitting-Bereich, estimated position, laserscanner, Sensoric, AT90CAN128

Abstract

To be able to determine one's exact position in a momentary environment is an important function of autonomous mobile systems. The way how the position coordinates are calculated, depends on the technic and the equipment of the autonomous robot. In this work I will introduce a model which makes it possible to determine the exact position coordinates of a mobile vehicle in a limited enviromentwith help of measuring data of the laserscanner and used sensors.

Danksagung

An dieser Stelle möchte ich mich bei allen Leuten bedanken, die mich während meines Studiums voll und ganz unterstützt haben und von denen ich viel Neues und Nützliches gelernt habe. Ich bin allen Professoren, Kommilitonen und Freunden sehr dankbar. Mein allergrößter Dank gilt meinen Eltern, meinem Bruder und meinen Verwandten, die mir mit aller Kraft geholfen haben und immer hinter mir standen.

Meine Bachelorarbeit war eine spannende, herausfordernde und interessante Zeit. Ich möchte hier bei meinem betreuenden Professor Dr. rer. nat. Reinhard Baran dafür bedanken, dass er immer mit Rat und Tat zur Verfügung stand und dafür, dass er mit seiner Freundlichkeit und Offenheit immer für die gute Atmosphäre in unserer Arbeitsgruppe sorgte. Ich danke außerdem meinen Kollegen Patrik und Thorsten, mit denen die Zusammenarbeit immer angenehm und konstruktiv war.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Einführung

Fahrerlose Fahrzeuge, intelligente Maschinen, autonome Roboter waren noch vor einigen Jahren der Bestandteil menschlicher Fantasie, den man nur in Sci-Fi-Bücher lesen und in den Hollywood-Filmen sehen konnte. Heute sind die Roboter ein unabdingbarer Teil unserer Realität geworden, die dem Menschen das Leben erheblich erleichtern. Sie machen sich heutzutage sowohl in militärischen, als auch in industriellen Bereichen zu Nutze, wobei die industrielle Nutzung der autonomen Roboter wegen ihrer Effektivität und Intelligenz mit jedem Jahr rasant steigt. Aus solcher Popularität der autonomen mobilen Roboter wurde die Robotik zu einer der bedeutendsten Forschungsrichtung, die in sich mehrere Erkenntnisse aus unterschiedlichen wissenschaftlichen Bereichen vereint.

Moderne autonome mobile Roboter sind komplexe, mechatronische, selbstregulierende Systeme, die in der Lage sind vielseitige Aufgaben von Aufklärungs- und Erkundungs-Missionen bis Transport- und Bergungs-Diensten, auszuführen. Je nach den Verwendungsansprüchen gibt es für sie verschiedene Ansätze bezüglich der Ausstattung mit speziellen Sensoren, Motoren, Elektronik und entsprechender Software. Die Arbeitsweise solcher autonomen Systeme spielt sich dabei nach einem gleichartigen Prinzip ab:

- Sensoren nehmen Eigenheiten der Umgebung wahr
- Messdaten werden von der Recheneinheit aufgenommen und ausgewährtet
- Entsprechende Funktionen werden durchgeführt und Aktoren kontrolliert

Um sich in der Einsatzumgebung zurechtzufinden, ist es für die autonomen Roboter von größerer Bedeutung ihre momentane Position zu bestimmen. Denn durch die Positionsbestimmung kann die Berechnung von Bahnkurven für die kollisionsfreie Fahrt zu einer bestimmten Zielposition wesentlich erleichtert werden. Im Allgemeinen versteht man unter Positionsbestimmung sowohl die aktuelle Position des jeweiligen mobilen Fahrzeugs, als auch die momentane Ausrichtung. Jotzo (2002) Davon abgesehen ist die Unterscheidung der Anforderungen an die Positionsbestimmung gerade für autonome mobile Systeme enorm, da ihr Einsatzgebiet sich von einer Zimmerfahrt bis auf die Erkundungsmissionen auf anderen Planeten erstreckt. Besondere Bedeutung macht hier die Unterscheidung zwischen lokalen und globalen Lokalisation. So kann es vorkommen, dass ein mobiles Fahrzeug seine globale Position kennt, kann sich dennoch in einem geschlossenen Raum, z.B. in einer Halle, nicht orten.

1.1 Problemstellung

Bei dem Einsatz eines fahrerlosen mobilen Fahrzeugs in einem von Außen abgeriegelten unbekannten Raum, wie im unterirdischen Tunnel oder in dem Kanalisationsschacht, oder in einem Gebäude, wäre es vorteilhaft und sinnvoll noch in der Startphase über die Umgebungskonturen und über die aktuelle Koordinatenposition des Fahrzeugs zu verfügen. Auf Grund der schon bekannten Konturen der Einsatzumgebung kann man die Positionslage besser schätzen und den weiteren Arbeitsverlauf des Roboters planen. Die unbekannten Umgebungsumrisse und somit die momentane Position kann man herausfinden, in dem man die Umgebung rund um das Fahrzeug mit Hilfe eines Laserscanner abtastet und aus den Laserscannerdaten dann die entsprechenden Positionskoordinaten und die Ausrichtung berechnet. Während der weiteren Fahrt, scannt man kontinuierlich oder in regelmäßigen Intervallen die Umgebung, um auf Basis der vorhandenen Ausgangsdaten die aktuellen Positionskoordinaten des autonomen Fahrzeugs zu ermitteln.

1.2 Zielsetzung

Das Ziel dieser Abschlussarbeit ist es einen Anpassungsalgorithmus für eine zuverlässige Positionsbestimmung eines autonomen mobilen Roboters in lokalen Anwendungen zu entwickeln. Dieses Algorithmus soll in der Lage sein, aus vorgelegten Sensoren- und Laserscanner-Daten exakte Koordinatenposition und Ausrichtungswinkel des Fahrzeugs in einem zweidimensionalen Raum zu berechnen.

Diese Arbeit enthält außer diesen noch vier weitere Kapiteln: Analyse, Entwurf, Evaluierung und Schluss. Der größte Teil des Stoffs des 2. Kapitels - Analyse setzt sich mit unterschiedlichen Positionsbestimmungstechniken auseinander und gibt die Anforderungen zu den eigenen Verfahren der Positionsbestimmung. Das nächste Kapitel ”Entwurf” beschreibt anfangs die verwendeten Hardware und grundlegenden Techniken, die für das Projekt wichtig sind, und gibt eine Vorstellung auf abstrakter Ebene über die Architektur des gesamten Konzepts. Danach werden die Aspekte der Realisierung und Softwareimplementierung erläutert. Im weiteren Kapiteln werden Verbesserungsmöglichkeiten und die Perspektiven des Konzepts in der Praxis angedeutet.

2 Analyse

Positionsbestimmungsverfahren werden nach zwei Betriebsarten klassifiziert, globale und lokale Positionsbestimmung. Das Ziel dieses Kapitels besteht darin, unterschiedliche Verfahren der Positionsbestimmung zu untersuchen und sie den konkreten Anwendungen zuzuordnen.

2.1 Positionsbestimmung in der Praxis

Wenn man mit Hilfe eines Positionsbestimmungsverfahrens seinen Standort und seine Orientierung berechnen will, steht er einer Reihe von verschiedenen möglichen Lösungsansätzen gegenüber. Dabei hängt die Wahl des Ansatzes von dem Maßstab des Anwendungsgebiets, wo die Positionsbestimmung stattfinden soll, ab. Für größere Flächen kommt die globale Positionsbestimmung ideal hin. Jedoch für kleinere Zone werden die globalen Positionskoordinaten nicht ausreichen. Da die heute eingesetzten globale Positionierungssysteme Abweichungen bei der Positionsgenauigkeit erweisen.

2.1.1 Techniken zur Positionsbestimmung

GPS

Ein Globales Positionierungssystem (GPS) ermöglicht die satellitengestützte Ortung von Objekten auf oder in der Nähe der Erde ABEL (2004). Zu einem der bekanntesten und am weitesten verbreiteten Globalen Positionierungssysteme gehört das in den 80-er Jahren von dem US-Verteidigungsministerium entwickeltes GPS mit dem Namen ”NAVSTAR”-(Navigation System with Time and Ranging). In den 90-er wurde das System für zivile Anwendungen freigegeben, jedoch mit Einschränkungen in der Genauigkeit. Im Jahr 2000 wurden jegliche Einschränkungen in der Genauigkeit abgeschaltet. GPS-”NAVSTAR” ermöglicht heutzutage für zivile Anwendungen mit bis auf 15 Meter genau das Standort zu lokalisieren, außerdem lässt sich die Geschwindigkeit und Bewegungsrichtung eines Objekts feststellen, was sich sehr gut fürs Auto-, Flugzeug- und Schiffverkehr erwiesen hat. GPS erfasst die Erde mit 24 nicht geostationären Satelliten, die die Erde in ca. 20.200 km Hohe umrunden. Das gestattet

auf jedem Platz der Erde zu jedem Zeitpunkt die nötige Anzahl von Satelliten verfügbar zu haben, die für die Positionsbestimmung notwendig sind. Auf diese Weise kann eine zweidimensionale Positionsbestimmung durch gleichzeitiges Empfangen der Signale von mindestens drei unabhängigen Satelliten erreicht werden, während für 3D-Positionsbestimmung Signale von vier und mehr Satelliten benötigt werden.(Sehe Abbildung 2.1)

Der Arbeitsprinzip globalen Navigationssysteme:

- Jedes Satellit sendet ein Datenpaket mit seiner Senderzeit und seiner aktuellen Position
- GPS-Empfänger vergleicht die Sendezeit und Empfangszeit von Satellitensignalen
- Aus der Zeitdifferenz wird die Entfernung der Satelliten berechnet
- Aus den Satellitenmessungen wird aktuelle Position durch Trilateration (Entfernungsmessung von drei Punkten aus) bestimmt

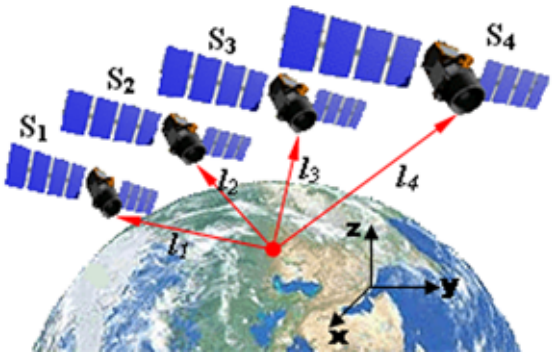


Abbildung 2.1: Globales Positionierungssystem (GPS)

Positionsberechnung erfolgt in einem globalen kartesischen Koordinatensystem. Die Entfernung zu dem Satelliten Δl kann man mit folgender Formel definieren.

$$s_i = \begin{pmatrix} x_s \\ y_s \\ z_s \end{pmatrix} \; e_i = \begin{pmatrix} x_e \\ y_e \\ z_e \end{pmatrix}$$

$$\Delta l = c_i * \Delta T_i$$

$$c_i * \Delta T_i = \sqrt{(x_{s_i} - x_{e_i})^2 + (y_{s_i} - y_{e_i})^2 + (z_{s_i} - z_{e_i})^2}$$

Darin sind s_i der Sendevektor eines Satelliten und e_i der Empfangsvektor. c_i gibt die Ausbreitungsgeschwindigkeit an, die aus GPS-Nachrichten berechnet wird, und ΔT_i gibt die Pseudolaufzeit eines Satellitensignals an[Jotzo:2001].

Außer dem amerikanischen Satellitennavigationssystem existiert noch ein weiteres von Russland entwickeltes globales Positionierungssystem ”GLONASS”, das allerdings nur der militärischen Kontrolle unterliegt. Hier darf man auch nicht das noch gebaute europäische Satellitennavigationssystem ”GALILEO” vergessen, das speziell für zivile Anwendungen konzipiert wird. Wann es allerdings fertig wird steht noch nicht fest.

WPS

Trotzt der Vielfalt der Geräte für globale Positionierungssysteme auf dem Markt sind sie aber wegen ihrer unzureichenden Genauigkeit für eine lokale bzw. interne Positionsbestimmung ungeeignet. Denn bei kleineren Räumen macht es kaum einen Unterschied, wo das Objekt bzw. ein mobiles Fahrzeug sich in der globalen Kontext befindet. In dem Fall muss das Fahrzeug seine Position in der augenblicklichen Umgebung mit anderen Mitteln herausfinden, um schließlich seine Primärfunktionen auszuführen.

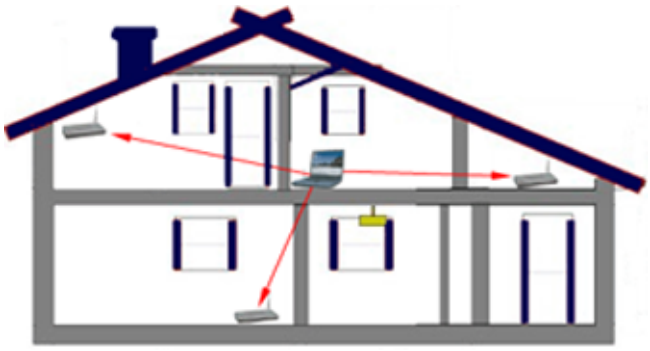


Abbildung 2.2: Wireless-LAN Positionierungssystem (WPS)

Zurzeit werden mehrere Projekte in Institutionen verschiedener Länder geführt, die sich mit dem Thema lokale Positionsbestimmung auseinander setzen. Einige Modelle zur lokalen Positionsbestimmung basierend auf Kurzstrecken-Funkstandards wie WLAN, Bluetooth oder RFID sind bereits entwickelt und erfolgreich getestet worden. Zum Beispiel, die Informatiker aus Berlin - Humboldt-Universität haben eine Methode gezeigt, die eine Positionslokalisierung innerhalb von Gebäuden per WLAN - (WPS) erlaubt. Dieses Verfahren beruht sowie GPS auf dem Trilaterationsprinzip. Es werden also zur Berechnung der Position von einem Objekt mindestens drei WLAN-Accesspoints benötigt. Eine speziell dafür entwickelte Software namens ” MagicMap” macht die Positionskalkulation und stellt dann die Position auf einer Karte dar, die allerdings selbst vorher erstellt werden muss. Die Entwickler der ” MagicMap” glauben, dass das Programm auch mit weiteren Funkstandarten, wie mobile Telekommunikationsnetze GSM und UMTS 5, Bluetooth oder RFID 5 ergänzt werden kann. Die Genauigkeit der WPS liegt im Vergleich zu GPS bei 3 bis 7 Meter. HU-Berlin (2008)

Landmarken

Prinzipiell kann man zu den alternativen Methoden der Positionsbestimmung greifen, wenn die vorbeschriebenen Techniken entweder nicht zugänglich sind, oder man auf die präzise Genauigkeit bei Positionslokalisierung angewiesen. Im Algemeinen kann es durch die Abtastung der momentanen Umgebung mit unterschiedlichen Sensorenarten wie Laser, Ultraschall, Infrarot oder sogar mit Kamera bewältigt werden. Der Entwickler entscheidet dabei welche Sensoren eingesetzt werden und nach welchem Berechnungsverfahren die Ermittlung der Position erfolgt. Als Hilfsmittel dafür können künstliche und natürliche Landmarken dienen. Die Landmarken sind bestimmte Punkte in der Umgebung die aus unterschiedlichen Positionen und Blinkwinkel erkennbar sind. Künstliche Landmarken sind Objekte, die speziell für die Positionsbestimmung in einer bekannten Umgebung platziert werden. Solche Landmarken können selber Signale ausstrahlen, und dementsprechend werden sie als aktive Landmarken charakterisiert. Da das Ziel dieser Abschlussarbeit die Positionsbestimmung in den unbekannten Räumen ist, werden hier die künstlichen Landmarken nicht weiter erläutert. Natürliche Landmarken sind die diversen Eigenheiten der Landschaft. In einem Zimmer oder einer Halle können das die Ecken, Wände, unterschiedliche Pfosten oder Heizungsgitter sein. Daher agieren sie als passive Positionspunkte. Die Positionsberechnung mit den natürlichen Landmarken erfordert rechen- und speicherintensive Erkennungsalgorithmen, was zu größten Nachteil wird.

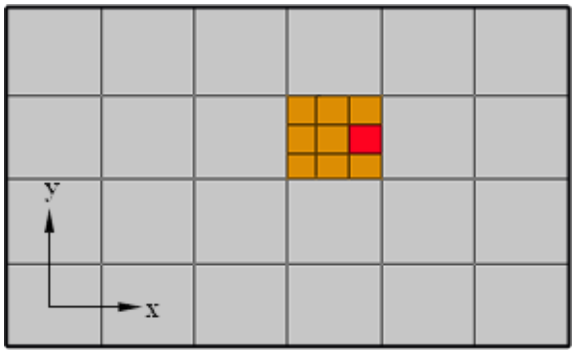


Abbildung 2.3: Occupancy Grid

Als Beispiel kann man einen rekursiven Erkennungsalgorithmus ”Occupancy Grid” nennen, der das Positionsbestimmungsproblem löst, indem er den Anwendungsraum in mehrere Quadrate zu einem Gitter zerlegt. Am Anfang werden alle Quadrate als leer oder nicht belegt initialisiert. Bei der Berechnung des Standortes werden die Quadrate einzeln nach der Belegung geprüft. Wenn ein der Quadrate als belegt bezeichnet wird, wird dieses Quadrat noch mal in mehrere Quadrate zerteilt. Und so wird es weiter vorgegangen, bis die gewünschte Auflösung erreicht wird. Die Abbildung 2.3 zeigt die visuelle Darstellung des Occupancy Grid.

Das Occupancy Grid - Verfahren stammt aus der Abteilung der Bilderkennung, hat sich aber mehr und mehr in der Robotik zur Positionsbestimmung verbreitet.

Ein weiteres Beispiel ist der Fitting-Algorithmus oder auf deutsch Anpassungsalgorithmus. Die Idee des Fittings besteht darin, den minimalen Abstand zwischen den Messpunkten und der Umrisslinie des Raums durch das Verschieben und Drehen der Messkoordinaten zu finden. Das Fitting-Verfahren ist der Zentralpunkt dieser Arbeit.

2.1.2 Beschränkungen und Schwächen einzelner Positionierungssysteme

	GPS	WPS	Laserscanner-PS
Messtechnik	Trilaterationsprinzip	Trilaterationsprinzip	Distanzmessung zum Polygon
Effektivität	Nicht ideal in geschlossenen, unterirdischen oder abgeschirmten, besonders aus Metall, Räumen	Beim Abstürz von einer Sendestation kann es zu dem Kollaps des gesammten System fuhren	Kann sowohl durch die Signale anderer Teilnehmer als auch durch die Strahl reflektierenden Oberflächen der Gegenstände irritiert werden
Genauigkeit	20-50 Meter	10-20 Meter	Bis auf Millimeter genau
Kosten	Enorm kostspielig und aufwendig, realisierbar nur auf staatlicher Ebene, fordert großes wirtschaftliches und technologisches Potenzial	Im Preis sehr hoch und etwas umständlich, allerdings kann von Unternehmen und technischen Institutionen aufgebaut und genutzt werden	Bezahlbar, zur Realisierung wird jedoch rechnerische Leistung des Computersystem gefordert

Tabelle 2.1: Schwächen der herkömlichen Positionierungsystemen

2.2 Anforderungsanalyse

2.2.1 Systemanforderungen

Anforderungen an Programmkonzept

- Alle Positionsberechnungen sollen in zweidimensionalen kartesischen oder in Polarkoordinatensystem erfolgen
- Der Algorithmus soll möglichst mit einfachen mathematischen Mitteln und Formeln erarbeitet werden
- Die fehlerhaften oder beschädigten Messwerte sollen bei der Auswertung berücksichtigt bzw. zur Fehlerbehandlung herangezogen werden
- Die Datenübertragung zwischen dem Scanner und dem Hauptrechner soll per Wireless-LAN erfolgen
- In der Initialisierungsphase soll die Ausgangsposition und die Ausrichtung des mobilen Fahrzeug bestimmt werden
- Aus den Sensorendaten soll der gefahrene Weg und somit die geschätzte Position des Fahrzeugs zu der Ausgangsposition errechnet werden
- Aus der geschätzten Position sollen im Verlauf des Algorithmus die exakte aktuelle Position des Fahrzeugs bestimmt werden

Softwareanforderungen

- Für gute Übersicht und verständliches Design soll die Software sowohl für Mikrokontroller, als auch für die Positionsbestimmung in einer Programmiersprache geschrieben werden (Programmiersprache C bzw. C++)
- Für die Kode-Programmierung und bei der Verwendung von externen Bibliotheken sollen vorzugsweise Open-Source oder Freeware-Komponente eingesetzt werden
- Der Programmkode soll verständlich, mit Kommentaren versehen, für die anderen freizugänglich und erweiterbar sein

Hardwareanforderungen

- Für normalen Algorithmusablauf muss der Hauptrechner genügend rechnerisches Potenzial vorweisen. (Speicherkapazität und Prozessorgeschwindigkeit)

- Kommunikation und Steuerung des autonomen mobilen Roboters soll per WLAN ablaufen
- Messdaten sollen berührungslos und räumlich hochauflösend erfasst werden
- Die beschädigten Teile des Fahrzeugs müssen leicht ersetzbar sein

2.2.2 Anwendungsfälle

Aufklärungsmissionen	Räume und Orte, die für die Menschen nicht zuganglich bzw. gefährlich sind (Gruben, Höhlen, Kanalisationsschachten Tunelle, Unfallgebiete, andere Planeten)
Transportdienste und Ablagerungsdienste	In der Industrie (große Hallen, Salle Flüre) für Bergungsaufgaben (Begrabene Räume, Keller)
Spiele	Bei Roboterspielen auf den künstlichen Flächen (Roboterfußball, Rally)

Tabelle 2.2: Mögliche Anwendungsbereiche der mobilen Roboter mit Positionsbestimmung

3 Entwurf

Bei dem Entwurf des Algorithmus zur Positionsbestimmung werden mehrere Ideen und mathematische Erkenntnisse implementiert, dessen Grundgedanken und Definitionen in diesem Kapitel eingehend erläutert werden.

3.1 Theoretische Grundlagen

3.1.1 Grundprinzip der lokalen Positionsbestimmung

Die Position und Ausrichtung eines autonomen mobilen Fahrzeugs kann in der eingegrenzten Anwendungsumgebung zu jeder Zeit und jeweils nach jeder gefahrenen Wegstrecke bestimmt werden. Dazu muss das Fahrzeug mit speziellen Sensoren, mit deren Hilfe die gefahrene Strecke berechnet werden kann, und der Laserscanneranlage, mit der das gesamte aktuelle Anwendungsgebiet abgetastet werden kann, ausgerüstet sein.

Der erste Schritt bei der Positionsbestimmung besteht jedoch darin, den Startpunkt und somit auch den Koordinatenursprung des mobilen Fahrzeugs festzustellen, bevor es sich in Bewegung setzt. Durch die Abtastung der Konturen des jeweiligen Raums mit dem Laserscanner kann man die Anfangsposition des Fahrzeugs bestimmen. Diese Anfangsposition entspricht dabei dem Koordinatenursprung des Fahrzeugs $P = (x_0; y_0)$, und seine Ausrichtung verläuft demzufolge parallel zur y-Achse. Erst nach der Bestimmung der Anfangsposition des Roboters ist er bereit zu fahren. Während der Fahrt des Roboters werden stetig seine Sensoren abgefragt, damit aus ihren Messdaten der nächste Positionspunkt bestimmt werden kann. Da es manchmal nicht möglich ist, bei den mathematischen Berechnungen alle Messdaten der Sensoren zu berücksichtigen (zum Beispiel: während die mathematische Berechnung läuft, liefern die Sensoren die neuen Messwerte, die nicht einbezogen werden können), darf man den Positionspunkt nach der gefahrenen Strecke nicht als richtige Position betrachten. Diese Position wird dann als geschätzte Position charakterisiert. In diesem Fall wird von der geschätzten Position aus eine weitere Raumscannung gemacht, um aus diesen Messdaten mit Hilfe von bestimmten mathematischen Algorithmen die exakten Fahrzeugkoordinaten zu ermitteln.

3.1.2 Koordinatensysteme

Der Position des mobilen Roboters in einem zweidimensionalen Raum kann eindeutig mit Hilfe von verschiedenen Koordinatensystemen bestimmt werden. Die Auswahl des Koordinatensystems hängt von der Bequemlichkeit der Durchführung der Berechnung. In Rahmen dieses Projekts für die Berechnung der Koordinaten eines Positionspsunktes in zweidimensionalem Raum kommen zum Einsatz kartesische und polare Koordinatensysteme, wobei die Werte bei der Kalkulation oftmals aus einem Koordinatensystem in ein anderes und umgekehrt umgerechnet werden müssen. Die Abbildung 3.1 zeigt die Darstellung der beiden Koordinatensysteme.

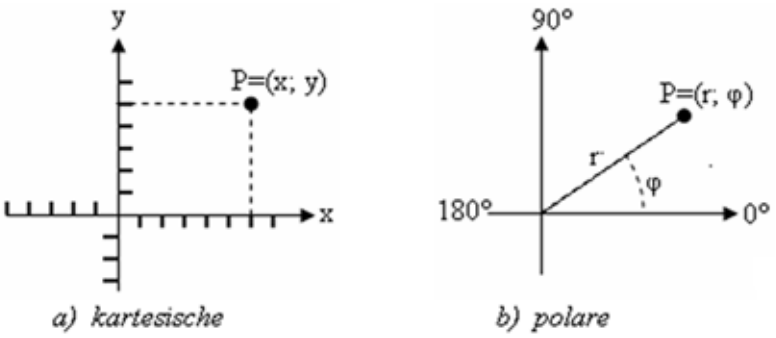


Abbildung 3.1: Kartesische und Polarkoordinatensystem

Kartesische Koordinaten zu Polarkoordinaten

Mit Hilfe folgender Formeln lassen sich alle Punkte von kartesischen Koordinaten in Polarkoordinaten umrechnen. Zunächst wird der Radius r nach dem Satz des Pythagoras berechnet.

$$r = \sqrt{x^2 + y^2}$$

Für die Berechnung des Winkels gibt es einige Möglichkeiten mit vielen Fallunterscheidungen. In Rahmen dieses Projekts wurde für die Positionsbestimmung die folgende Arkustangens-Funktion genommen, da die mit weniger Fallunterscheidungen auskommt.

$$\varphi = \begin{cases} 2 \arctan \frac{y}{r+x} & \text{für } r + x \neq 0 \\ \pi & \text{für } r + x = 0 \end{cases}$$

Bei der Berechnung der Verschiebung der Koordinaten eines Punktes hat sich das kartesische Koordinatensystem am effektivsten erwiesen. Dabei addiert man einfach die verschobenen x_v- und y_v- Werte mit den Anfangskoordinaten des Punktes zusammen.

$$x = x_0 + x_v$$

$$y = y_0 + y_v$$

Polarkoordinatensystem

Das Polarkoordinatensystem erweist sich sehr hilfreich bei der Berechnung einer Richtungsdrehung . Dabei werden der Anfangswinkel und der gedrehte Winkel zusammenaddiert $\varphi = \varphi_0 + \varphi_d$. Ebenfalls können die Polarkoordinaten leicht in die kartesischen Koordinaten umgewandelt werden. Die Umrechnung der Polar- in kartesische Koordinaten zeigen folgende Formeln:

$$x = r * \cos\varphi$$

$$y = r * \sin\varphi$$

3.1.3 Fittingverfahren bei der Positionsbestimmung

Das Grundprinzip des Fittingverfahrens bei der Positionsbestimmung für autonome mobile Systeme besteht darin, einen minimalen Abstand zwischen den von der geschätzten Position aus gemessenen Polygonpunkten zu dem Ausgangspolygon bzw. zu der eigentlichen Umgebungskontur durch sukzessive Winkeldrehung und Koordinatenverschiebung zu bestimmen. Für die Implementierung des Fitting-Models werden folgende Parameter benötigt:

- Ausgangskoordinaten des Fahrzeugs
- Ausgangskontur der momentanen Umgebung (Polygon)
- Geschätzte Positionskoordinaten und Ausrichtung des Fahrzeugs nach der Fahrt
- Distanzpunkte zur Umgebungskontur von der geschätzten Position aus

Die Berechnung der geschätzten Positionskoordinaten des Fahrzeugs und Bestimmung des Polygons werden in unteren Kapiteln erläutert.

Angenommen, dass alle oben beschriebe Parameter vorhanden sind, kann das Fitting-Model nach folgender Vorgehensweise realisiert werden:

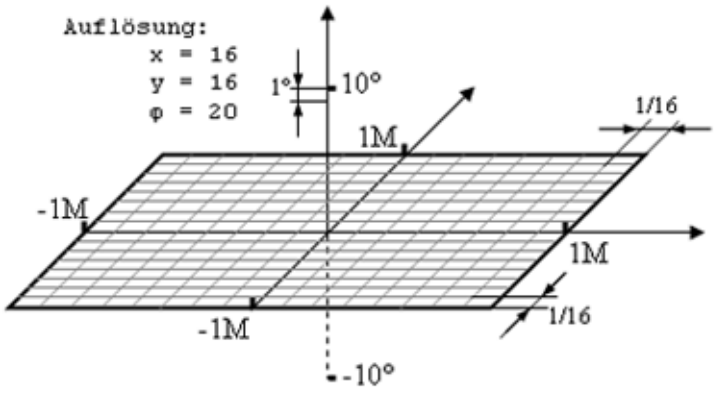


Abbildung 3.2: Fitting-Bereich

- An erster Stelle werden die geschätzten Positionskoordinaten zu dem Koordinatenursprung zurück verschoben und um den Richtungswinkel zurückgedreht, so dass die geschätzte Ausrichtung mit der Ausgangsausrichtung übereinstimmt. Demzufolge werden auch alle von der geschätzten Position aus mit dem Laserscanner gemessene Distanzwerte bezüglich der Ursprungskoordinaten umgerechnet.
- Im nächsten Schritt werden Fitting-Bereiche für x- und y Koordinaten und für den φ Richtungswinkel bestimmt. Fitting-Bereiche sind spezielle Berechnungsräume, wo die Koordinatenverschiebung und Winkeldrehung des Mittelpunkts der geschätzten Distanzwerten stattfinden. Für die provisorische, visuelle Darstellung solcher Fitting-Bereichen für x-, y-Koordinaten und den φ-Winkel, die die folgende Abbildung 3.2 zeigt, wurden die Größen der Fitting-Bereichen folgendermaßen ausgewählt: bei x und y von -1 bis 1 Meter beim φ von -10° bis 10°. Für normale Positionsbestimmung hängen die Größen der Fitting-Bereichen für x-, y-Koordinaten und den φ-Winkel jedoch von der Größe der Abweichung zwischen der geschätzter und exakter Position. Die Abweichungen variieren sehr, wenn das Fahrzeug geradeaus oder eine Kurve fährt. Das liegt daran, dass keine genaueren Daten von der Lenkung und Geschwindigkeit für die Berechnung zur Verfügung stehen. Schreibern (2007) Die Messungen bei der geraden Fahrt ergaben die Abweichung in der Position 5-6$^0/_0$, wobei bei einer Kurvenfahrt lag die Abweichung bei 15-20$^0/_0$. In Rahmen dieses Projekts werden die Fitting-Bereiche für x-, y-Koordinaten und den φ-Winkel unter der Annahme, dass die Abweichung bei 25$^0/_0$ liegt, speziell so groß genommen, damit alle mögliche Bewegungsfälle des Fahrzeugs abgedeckt werden.
- Als Nächstes wird eine Auflösung für x, y und φ Fitting-Bereiche gewählt. Die Auflösung bestimmt die Größe der Schritte, nach denen eine Berechnung des mittleren Abstandes durchgeführt werden soll, und somit auch die Genauigkeit bei der Positionsbestimmung. Folgende Tabelle 3.1 gibt die Anzahl der Kombinationen aller x, y und φ

Auflösung (Schrittenanzahl)	Genauigkeit beim Fitting-Bereich $2M * 2M * 20°$	Anzahl möglicher Konfigurationen
$x = 40$ $y = 40$ $\varphi = 20$	50mm und 1° bei der Ausrichtung	$40^2 x 20 = 32000$
$x = 60$ $y = 60$ $\varphi = 40$	33mm und 0,5° bei der Ausrichtung	$60^2 x 40 = 144000$
$x = 80$ $y = 80$ $\varphi = 10$	25mm und 2° bei der Ausrichtung	$80^2 x 10 = 64000$

Tabelle 3.1: Anzahl der Konfigurationen bei unterschiedlichen Auflösungen

Koordinaten und relative Genauigkeit für unterschiedlichen Auflösungen an. Vor jeder Berechnung werden alle Messwerte von der geschätzten Position der entsprechenden Koordinatenkombination angepasst.

- Um den mittleren Abstand der Messpunkte von einer x, y und φ Koordinatenkombination zu dem Polygon zu bestimmen, werden zuerst alle Abstände der einzelnen Punkte berechnet, miteinander summiert und anschließend durch ihre Anzahl geteilt. Die Formel dafür sieht wie folgt aus:

$$\text{mittlerer Abstand} = \frac{\sum_{i=1}^{N} x_i}{N}$$

- Im nächsten Schritt wird der gefundene mittlere Abstand bezüglich dieser Koordinatenkombination gespeichert. Die Abbildung 3.3 stellt ein Paar möglicher Situationen dar, wie sich die Abstände bei unterschiedlichen x, y und φ Kombinationskoordinaten ändern.
- In der Abschlussphase, nachdem alle Abstandsberechnungen für alle x, y und φ Kombinationen durchgeführt und gespeichert worden sind, startet ein Vorgang, bei dem der minimale Abstandswert und somit die richtige Position gesucht wird. Nach der Bestimmung des minimalen Abstands werden die Korrekturen an den geschätzten Positionskoordinaten vorgenommen.

3.1.4 Geschätzte Position- und Streckenbestimmung

Im Algemeinen beginnt das autonome mobile Fahrzeug seine Fahrt von seinem Koordinatenursprung. Die Fahrstrecke kann dabei sowohl geradlinig als auch gekrümmt verlaufen.

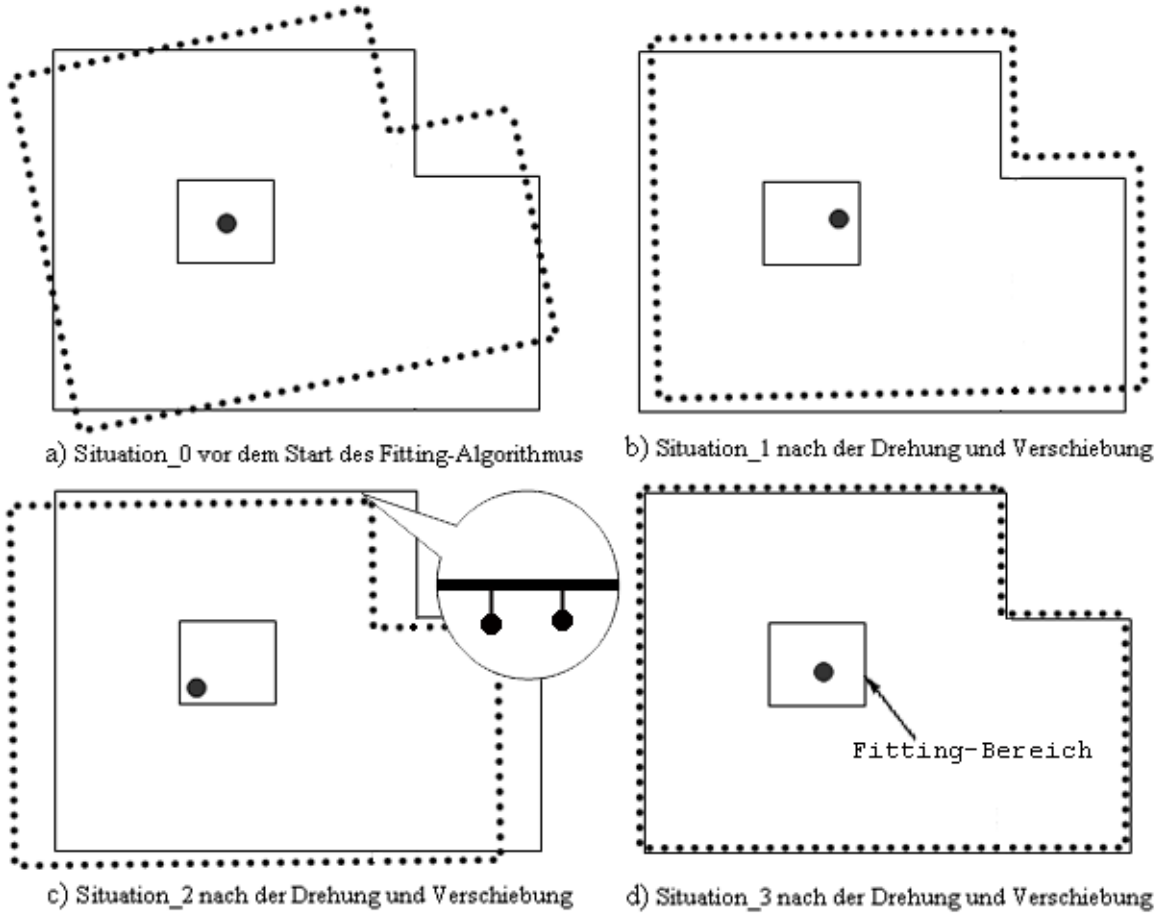


Abbildung 3.3: Mögliche Situationen bei der unterschiedlichen $X-$, $Y-$ und $\varphi-$ Koordinatenkombinationen

Die Richtung bei der geraden Fahrt hängt von der derzeitigen Ausrichtung des mobilen Fahrzeugs ab. Die Berechnung des zurückgelegten Weges s erfolgt nach folgender Formel, wo v die Geschwindigkeit des Fahrzeugs und t die Zeit sind, die das Fahrzeug für die Streckenfahrt gebraucht hat.

$$s = v * t$$

Bei mobilen Robotern kann man die gefahrene Strecke mit anderen unterschiedlichen Methoden herausfinden. Eine Methode beruht auf folgendem Prinzip: $s = l_r * U$ Die bekannte Kreislänge des Rades des Fahrzeugs l_r wird mit Radumdrehungen multipliziert. Die Radumdrehungen können bei einem autonomen mobilen Fahrzeug durch optischen Sensor überwacht werden. Die zweite Methode, die in Rahmen dieses Projekts angewendet wird, basiert auf der Drehzahlmessung des Motors des Fahrzeugs in Abhängigkeit von dem angelegten Gang. Folgende Tabelle gibt die gemessene Geschwindigkeit bei unterschiedlichen Gängen und gleicher Motordrehzahl.

Gang	Motordrehzahl U/min	Geschwindigkeit mm/s
1. Gang	200	721
2. Gang	200	1435
3. Gang	200	2826

Tabelle 3.2: Streckenmessung mit glecher Motordrehzahl bei unterschiedlichen Schaltgängen

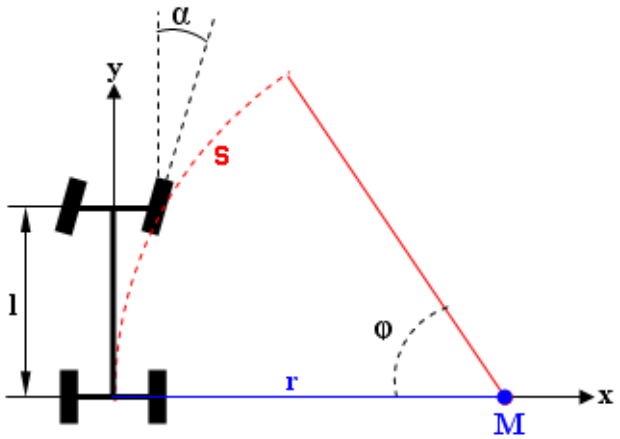


Abbildung 3.4: Positionsbestimmung bei einer Kurvenfahrt

Wenn die Lenkachse des Fahrzeugs einen Lenkwinkel α aufweist und der Winkel sich einige Zeit während der Fahrt nicht ändert, dann dehnt sich die Fahrstrecke s des mobilen

Fahrzeugs in genau dieser Zeitphase im Kreisbogen um einen festen Punkt M mit einem bestimmten Radius r aus. Die folgende Skizze 3.4 zeigt die beschriebene Situation. Der Radius r lässt sich mit folgender trigonometrischen Formel berechnen:

$$r = \frac{\text{Radabstand}}{\text{Lenkwinkel}} = \frac{l}{\tan\alpha}$$

Da die Hinterräder nicht lenkfähig sind, liegt der Fahrkreismittelpunkt M auf der Verlängerung der Hinterachse des mobilen Fahrzeugs. In der auf der Skizze dargestellten Situation befindet sich das mobile Fahrzeug in seiner Startposition. Der Lenkwinkel der vorderen Achse α bestimmt dabei die Richtung des Mittelpunkts M auf der $x-$Achse.

$$\begin{pmatrix} \text{Bei positiven} & \alpha & M = (-r, 0) \\ \text{Bei negativen} & \alpha & M = (r, 0) \end{pmatrix}$$

Wenn der Radius und die Fahrstrecke bekannt sind, lässt sich die neue Position wie folgt ausrechnen: Als erstes findet man den Winkel ϕ der gefahrenen Strecke zu den Kreismittelpunkt M

$$\phi = \frac{s * 360^\circ}{2 + \pi * r}$$

Danach rechnet man wie folgt die ”geschätzten” Positionskoordinaten:

$$P = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} * \begin{pmatrix} r \\ 0 \end{pmatrix} - \begin{pmatrix} r \\ 0 \end{pmatrix} = \begin{pmatrix} x = \cos\phi * r - r \\ y = \sin\phi * r - 0 \end{pmatrix}$$

3.1.5 Vektorrechnungen für Polygondefinition

Während der Scannung eines Raums mit der eingegebenen regelmäßigen Winkelauflösung liefert der Laserscanner Messwerte von zu jedem Winkel passenden Abständen. Auf Basis dieser Messdaten kann man eine Umrisslinie eines Raums darstellen, indem man alle Punkte miteinander in eine geschlossene Linienkette verbindet. Mathematisch bezeichnet man dieses Verfahren als Definition eines Polygons. Hierbei sind die Messpunkte seine Ecken und die Linien zwischen den Punkten sind seine Kanten. Je größer die Anzahl von Messpunkten ist bzw. je größer die Winkelauflösung beim Laserscanner eingestellt wird, desto detaillierter wird das Polygon zur eigentlichen Umrisskennlinie des Raums. Um Polygon zu bestimmen, muss man die Länge und Richtung jedes einzelnen Abschnitts zwischen zwei Punkten herausfinden. Mit folgender Vektormathematik lässt sich die Abschnittlänge berechnen.

Berechnung eines Abstandsvektors

Alle Rechenoperationen der Vektoren und ihre Darstellung werden in einem zweidimensionalen, kartesischen Koordinatensystem durchgeführt. Dabei besteht ein Vektor aus der Summe zweier Vektorkomponenten bzw. aus seinen Projektionen auf die beiden Koordinatenachsen: $\vec{a} = \vec{a}_x + \vec{a}_y$. Jeden Vektor kann man außerdem durch seinen Einheitsvektor $\vec{e}$ darstellen, der die Richtung und die Länge von 1 des eigentlichen Vektors $\vec{a}$ präsentiert. Die Formel zur Berechnung des Einheitsvektors in der Komponentendarstellung lautet somit:

$$\vec{e}_a = \left(\frac{\vec{a}_x + \vec{a}_y}{|\vec{a}|} \right)$$

Da alle vom Laserscanner gemessenen Abstände bekannt sind, kann man jeden von denen eventuell als eine Vektoreinheit darstellen. Dieser vom Nullpunkt zu einem Messpunkt gerichtete Vektor wird dann als Ortsvektor bezeichnet, der als Summenvektor aus Koordinaten x und y darstellbar ist. Zwischen den Endpunkten zweier benachbarten Ortsvektorn befindet sich der gesuchte Abstandsvektor (Sehe Abbildung 3.5 a), der, im Grunde genommen, nicht anderes als ein Differenzvektor der beiden Ortsvektorn ist. Die Berechnung des Differenzvektors erfolgt daher nach der Subtraktionsregel von Vektoren.

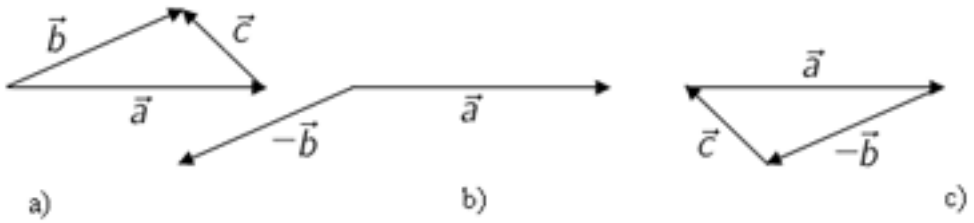


Abbildung 3.5: Subtraktion der Vektoren

- Der Vektor $\vec{b}$ wird zunächst in seiner Richtung umgekehrt: Dies führt zu dem inversen Vektor -$\vec{b}$. (Abbildung 3.5 b)
- Dann wird der Vektor -$\vec{b}$ parallel zu sich selbst verschoben, bis sein Anfangspunkt in den Endpunkt des Vektors $\vec{a}$ fällt. (Abbildung 3.5 c)
- Der vom Anfangspunkt des Vektors $\vec{a}$ zum Endpunkt des Vektors -$\vec{b}$ gerichteter Vektor ist der gesuchte Differenzvektor $\vec{c} = \vec{a} - \vec{b}$ Papula (1998)

In Vektorkomponentendarstellung sieht das dann wie folgt aus:

$$\vec{c} = \vec{c}_x + \vec{c}_y$$
$$\vec{c}_x = \vec{a}_x - \vec{b}_x$$

$$\vec{c}_y = \vec{a}_y - \vec{b}_y$$

Nun sind die Koordinaten des Differenzvektors bekannt. Als nächstes berechnet man die Länge des Differenzvektors, indem man sein Betrag herausfindet. Den Betrag erhält man unmittelbar aus dem Satz des Pythagoras, indem eine Quadratwurzel aus der Summe der Quadraten von x und y Koordinaten gezogen wird.

$$|\vec{c}| = \sqrt{c_x^2 + c_y^2}$$

Berechnung eines Abstands zwischen einem Polygonabschnitt und einem Punkt

Der zentrale Schwerpunkt des Positionsbestimmungsalgorithmus ist die Suche nach den Abständen zwischen den von der geschätzten Position gemessenen Punkten und den eigentlichen Polygonkanten. Um die Abstände zwischen Messpunkten und dem Polygon zu ermitteln geht man nach einer Mehrschritttaktik vor.

- Es wird zuerst der Vektor $\vec{p}$ von dem Anfangspunkt des Polygonsegments zu dem Messpunkt errechnet
- Als nächstes wird das Skalarprodukt zweier Vektoren den Polygonabschnittsvektors und Vektor $\vec{p}$ gesucht, bzw. Projektion vom Vektor $\vec{p}$ auf den Polygonabschnitt
- Wenn die Projektion auf der Polygonkante liegt, ermittelt man den Kreuzprodukt zweier Vektoren um den Flächeninhalt des von beiden Vektoren aufgespannten Parallelogramms zu erhalten
- Aus den Flächeninhalt des Parallelogramms wird die Höhe berechnet, die der eigentliche Abstand ist

Bestimmung eines Vektors durch zwei Punkte

Der Vektor $\vec{p}$ ist ein durch zwei Punkte festgelegter Vektor. Es handelt sich dabei um einen Anfangspunkt $P_1 = (x_1; y_1)$, was dem Anfang des Polygonabschnittsvektors entspricht, und einen Endpunkt $P_2 = (x_2; y_2)$, der einem Messpunkt der geschätzten Position entspricht.(Sehe die Abbildung 3.6) Sind die beiden Punkte des Vektors bekannt, so lautet seine Komponentendarstellung wie folgt:

$$\vec{p} = \overrightarrow{P_1 P_2} = (\vec{x}_2 - x_1)\vec{e}_x + (y_2 - y_1)\vec{e}_y = \begin{pmatrix} x_2 - x_1 \\ y_2 - y_1 \end{pmatrix}$$

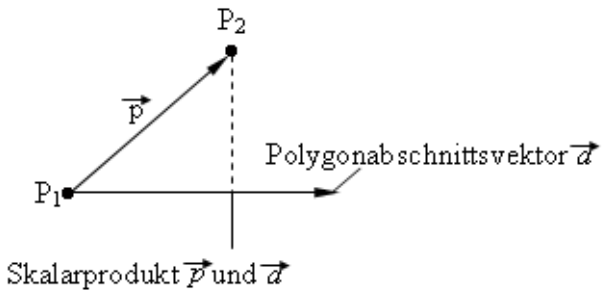


Abbildung 3.6: Skalarprodukt

Skalarprodukt zweier Vektoren

Das Skalarprodukt zweier Vektoren, eines Polygonabschnittsvektors und des Vektors $\vec{p}$, zeigt eine Projektion des Vektors $\vec{p}$ auf den Polygonabschnittsvektor.(Sehe die Abbildung 3.6) Das Skalarprodukt λ zweier Vektoren des zweidimensionalen euklidischen Raums wird als die Summe der Produkten der Komponenten der beiden Vektoren definiert und lässt sich nach folgender Formel berechnen:

$$\vec{a} * \vec{b} = \begin{pmatrix} a_x \\ a_y \end{pmatrix} * \begin{pmatrix} b_x \\ b_y \end{pmatrix} = a_x b_x + a_y b_y = \lambda$$

Kreuzprodukt zweier Vektoren

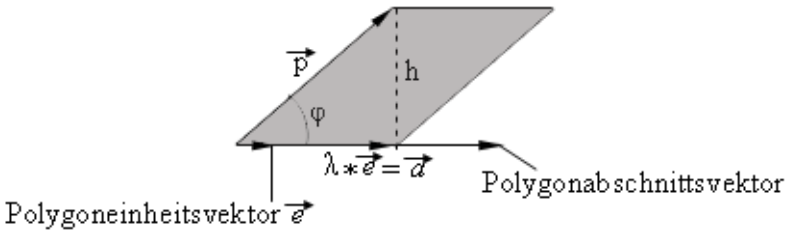


Abbildung 3.7: Kreuzprodukt zweier Vektoren

Wenn man die Skizze 3.7 betrachtet, fällt es sofort auf, dass der gesuchte Abstand die Höhe eines Parallelogramms ist, wobei der Vektor $\vec{p}$ seine Seitenlinie und $lambda * \vec{e} = \vec{a}$ seine Grundlinie sind. Den Flächeninhalt A des von den Vektoren $\vec{p}$ und $\vec{a}$ aufgespannten Parallelogramms erhält man nach folgender Formel, wo A - die Flächeninhalt, a - Grundlinie, h - Höhe und p - Seitenlinie sind.

$$A = a * h = a * p * \sin\varphi = \mid \vec{a} \mid * \mid \vec{p} \mid * \sin\varphi$$

Dies ist aber genau der Betrag des Vektorproduktes $\vec{p} \times \vec{a}$. Das Vektorprodukt $\vec{p} \times \vec{a}$ ist eine vektorielle Größe und wird als Kreuzprodukt der Vektoren $\vec{p}$ und $\vec{a}$ bezeichnet. Papula (1998) Realisiert wird das Kreuzprodukt nach volgender Formel:

$$\vec{p} \times \vec{a} = \begin{pmatrix} \vec{p}_x \\ \vec{p}_y \end{pmatrix} * \begin{pmatrix} \vec{a}_x \\ \vec{a}_y \end{pmatrix} = \vec{a}_x * \vec{p}_y - \vec{p}_x * \vec{a}_y$$

Nach dem die Fläche des Parallelogramms bekannt ist, kann man schnell wie folgt seine Höhe und damit den gesuchten Abstand berechnen.

$$h = \frac{A}{\vec{a}}$$

3.2 Beschreibung der vorhandenen Hardware und Bestandteile

3.2.1 Mobile Plattform

Abbildung 3.8: LKW - ”WEDICO”

Als Basis für eine mobile Plattform wurde ein 3-achsiges Model eines LKW-Zugwagens von Firma ”WEDICO” mit einem offenen Auflieger in Maßstab 1:16 genommen, wie auf der Abbildung 3.8 gezeigt wird. Das LKW-Model ist mit einem Elektromotorantrieb, 2-stufigen Standartgetriebe, 3-Gang-Schaltgetriebe, Motorgeräuschimitationsgenerator und Beleuchtungssystem ausgestattet. Vorne an der Stoßstange und seitlich an dem Fahrgestell der Zugmaschine sind die Sharp-Abstandssensoren platziert. Die gesamte Fahr-, Lenk- und Betriebs-Funktion des Fahrzeugs wird vom AVR - AT90CAN128 Mikrokontroller gesteuert, der sich im hinteren Teil der Fahrerkabine befindet. Unter dem Mikrokontroller wurde ein Seilzugsensor befestigt, der bei einer Kurvenbewegung des Fahrzeugs den Winkel zwischen der Zugmaschine und dem Auflieger kontrolliert. Oben auf der Kabine ist ein Display für Debuging-Ausgaben angebracht. Im vorderen Teil des Aufliegers ist der Laserscanner auf einer hohen Plexiglasplattform befestigt, damit keine Fahrzeugelemente in den Blickwinkel des Laserscanners kommen. Zur Steuerung des Laserscanners sitzt daneben ein weiterer Mikrokontroller vom gleichen Typ. Beide Mikrokontroller sind durch getrennte CAN-Busse mit einem WLAN - Kommunikationsmodul (IGW400/CAN) verbunden, das sich inklusive einer Antenne in der Mitte des Aufliegers befindet. Zusätzlich trägt der LKW als Stromversorgung für die Motore, Mikrokontroller und den Laserscanner eine leistungsstarke Akkubatterie.

Das LKW-Modell kann sich sowohl autonom als auch kontrolliert bewegen. Bei kontrollierter Fahrt nutzt das Fahrzeug spezielle Steuerungsbefehle, die er von einem Kontroller oder einem Steuerungsgerät per WLAN empfängt. Diese Befehle bestimmen die Lenkung (Drehwinkel der Lenkmotoren) und Geschwindigkeit (Motordrehzahl und Schaltgänge) des Wagens. Bei der autonomen Fahrt werden die Fahrrichtung und Schnelligkeit auf Basis der von den Abstandsensoren gelieferten Messdaten berechnet.

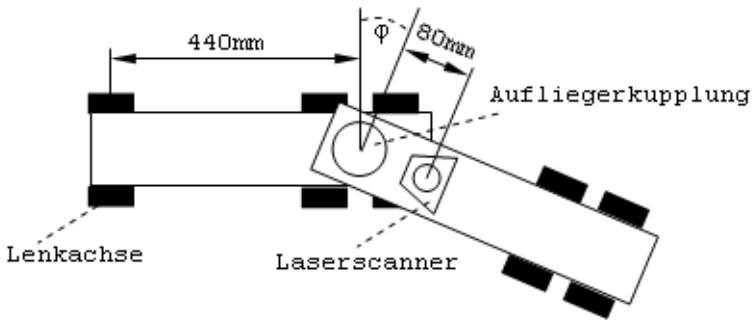


Abbildung 3.9: Abstände zwischen Vorderachse, dem Drehpunkt und Lasescanner

Bei dem LKW-Model bestimmt die vordere Lenkachse die Fahrrichtung. Deshalb ist es von größerer Bedeutung, dass während der Einsatzfahrt des LKWs alle Raummessungen vom Laserscanner bezüglich der vorderen LKW- Lenkachse gemacht werden. Da aber der Laserscanner auf dem Auflieger platziert ist, sollen alle Messwerte des Laserscanners bezüglich der Lenkachse umgerechnet werden. Dafür werden Abmessungen der Abstände von der

Lenkachse zur Aufliegerkupplung und von der Aufliegerkupplung zu dem Laserscanner gemacht und als festen Größen gespeichert. Außerdem muss man beachten, dass der Auflieger während der Kurvenfahrt sich dreht. Dadurch ändert sich der Winkel zu dem Zugwagen. Solche Situation wird auf der folgenden Skizze 3.9 dargestellt.

Der Algorithmus für die Konvertierung der Messpunkte vom Laserscanner bezüglich der Lenkachse wird nach folgender Reihenfolge durchgeführt: (dazu sehe die Abbildung 3.10)

- Die Koordinaten des Laserscanners und seiner Messpunkten werden erst nach dem Abstand (Laserscanner $\rightarrow$ Aufliegerkupplung) der y_1 Achse entlang verschoben.
- Dann werden die Koordinaten um den Winkel φ gedreht.
- Schließlich werden die Koordinaten nach dem Abstand (Aufliegerkupplung $\rightarrow$ Lenkachse) der y_2 Achse entlang verschoben.

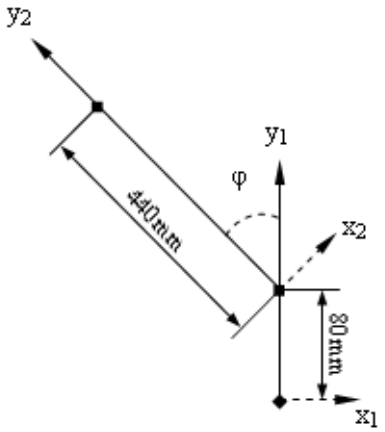


Abbildung 3.10: Konvertierung der Messpunkte bezüglich der Lenkachse

3.2.2 Lasermesssystem LD-OEM

Ein Lasermesssystem LD-OEM eignet sich für Anwendungen, bei denen präzise, berührungslose Kontur- und Umgebungsmessungen in zweidimensionalen Polarkoordinaten gefordert sind. Dieses Messsystem arbeitet nach dem Prinzip ”Pulslaufzeitmessung”, bei der die Entfernung zum Objekt aus der Laufzeit berechnet werden, die die gepulste Laserstrahle von der Aussendung bis zum Empfang der Reflektion im Sensor benötigen.

In Rahmen dieser Arbeit, für die Raumvermessungsaufgaben wurde ein $LD-OEM1000$ Laserscanner von der Firma ”SICK AG” verwendet.(sehe Bild 3.11). Dieser Laserscanner arbeitet mit einem Infrarotlicht-Laser der Klasse 1 und einer Wellenlänge $\lambda = 905nm$. Die

Abbildung 3.11: Laserscanner LD-OEM1000

austretende Strahlung ist im normalen Betrieb für die menschliche Augen und die menschliche Haut ungefährlich. Zudem ist das LD-OEM so konstruiert, dass es die Umwelt so wenig wie möglich belastet.

Der Laserscanner hat ein Aluminiumgehäuse mit einem oben angebrachten drehbaren Scannerkopf mit einem elektro-optischen Sensor, der die kreisförmige Abtastung mit einer wählbaren Frequenz von 5 bis 20 Hz und einem wählbaren Sichtbereichs von 0° bis 360° ermöglicht. Dabei werden fortlaufend jeweils nach einem einstellbaren Winkelschritt ein bis maximal 14,4 kHz einstellbares Laserpuls und damit eine Entfernungsmessung ausgelöst. Die maximale Auflösung der Winkelschrittweite liegt bei 0,125°. Der Nullpunkt befindet sich,wie die Abbildung 3.12 zeigt, im hinteren Teil des LD-OEM. Die Reichweite des Scanners ist von der Umgebungsoberfläche abhängig. Je besser die Oberfläche die auftreffende Strahlung reflektiert, umso größer ist die Reichweite. Bei normalen Lichtverhältnissen und 90% Reflektion beträgt die Reichweite ca. 100 m.

Das Gerät arbeitet im normalen Betrieb mit einer Versorgungsspannung von DC $24V \pm 20\%$. Hierbei entsteht im Gerät eine Spannung von maximal 270V. Der Energiebedarf des Scanners ist von seiner Betriebsart abhängig. Beim Einschalten ist es 36W (1,5A) und in Dauerbetrieb 12W (0,5A) zusätzlich max.12W beim je verwendeten Schalteingang. Der Laserscanner verfügt über vier Betriebsmodis:

- IDLE-Modus: Standby (Scannerkopf in Ruhe, Laser aus)
- ROTATE-Modus: Rotation (Scannerkopf dreht, Laser aus)
- MEASURE-Modus: Messbetrieb
- ERROR-Modus: Fehlerabfrage

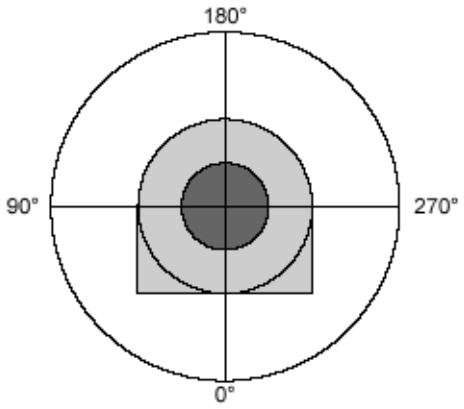


Abbildung 3.12: Winkelachsendarstellung des LD-EOM

Den Status des Geräts kann man an 4 LEDs erkennen, die an vorderer Seite des Gehäuses angebracht sind. Der Laserscanner ist mit drei Datenschnittstellen CAN, Ethernet und umschaltbare RS422 und RS-232 ausgerüstet und kann somit an ein Rechnersystem angeschlossen werden. Über die Datenschnittstelle kann das Gerät mit PC-Software ”LD-CONFIG” in seinen Grundfunktionen (Datenkommunikation, Scannbereich, Winkelschritte) parametriert und sein Betrieb gestartet bzw. gestoppt werden. Auf Anforderung gibt das LD-LRS2100 die Rohdaten der gemessenen Profile in zweidimensionalen Polarkoordinaten fortlaufend als Hex-Werte aus.

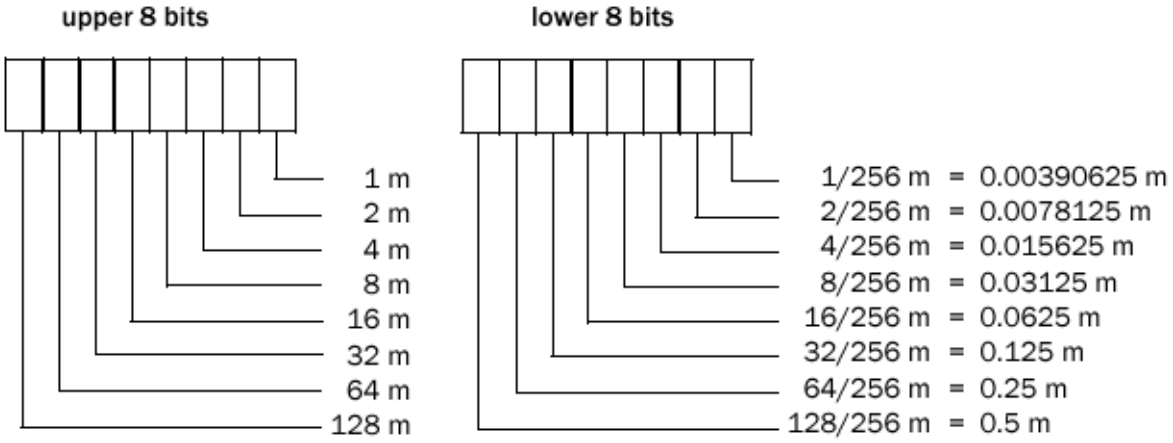


Abbildung 3.13: Binäre Darstellung eines 16-Bit Distanzmesswertes

In der parallel geführten Diplomarbeit von Patrik Schlaak wurden die Kommunikation per WLAN- Modul mit dem Laserscanner und Versendung und Empfang von Datenpaketen realisiert. Ein Datenpaket enthält alle Distanzmesswerte von einer Rundmessung. Die Abbildung

3.13 zeigt die binäre Darstellung eines Distanzmesswertes. Und die Abbildung 3.14 zeigt die Messwerte mit ihren Winkeln nachdem sie empfangen und in Dezimalwerte konvertiert wurden. Diese Messdaten weisen aber aus noch bisher unerklärlichen Gründen einige Messfehler auf. Deshalb werden sie, bevor sie für die Positionsbestimmung eingesetzt werden, der Fehlerkorrektur unterzogen, indem der Mittelwert eines Distanzmesswertes von mehreren Paketen berechnet wird.

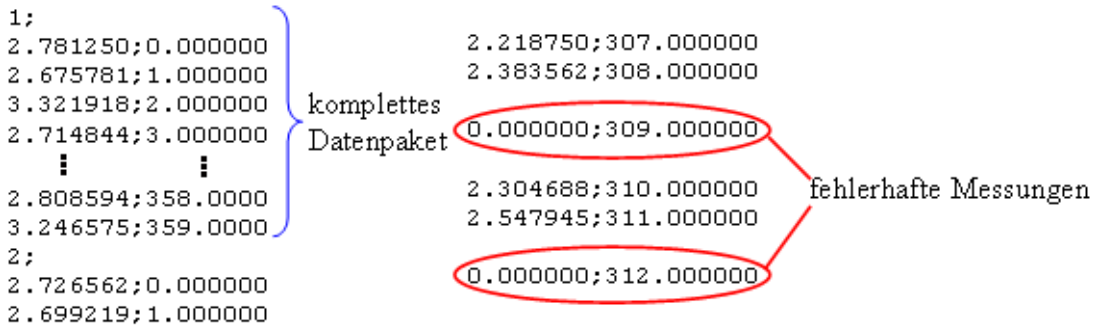


Abbildung 3.14: Komplettes Datenpaket und fehlerhafte Messwerte

3.2.3 Seilzugsensor

Bei einer Raumvermessungsfahrt geht man davon aus, dass alle Messwerte bezüglich der Vorderlenkachse des in Kapitel 2.1.1 beschriebenen Fahrzeugs sein müssen. Da die Laserscanneranlage auf dem Auflieger montiert ist, werden alle Messdaten und somit die aktuelle Koordinateposition bezüglich des Laserscanners aufgefasst. Deshalb müssen die Messwerte der Vorderachse des Fahrzeugs angepasst werden, bevor man eine Positionsbestimmung durchführt. Dabei muss man den Winkel, der beim Fahren einer Kurve oder eines Abbiegmanövers zwischen dem Zugwagen und seinem Auflieger entsteht, und die konstanten Abstände von der Vorderachse bis Anhänger-Kupplung und von der Anhänger-Kupplung bis Laserscannerkopf berücksichtigen.

Um den Drehwinkel des Aufliegers aufzunehmen, wurde ein Seilzugmesser der Baureihe SL00 in Low-Cost Ausführung mit eingebautem FSG-Drehgeber verwendet. Der verwendete Seilzugsensor SL00 125 GS55, der auf der Abbildung 3.15 dargestellt ist, hat eine flache, gewichtsarme Gehäusebauform aus Kunststoff mit vielfältigen Befestigungsmöglichkeiten und enthält massearme hochgenaue Messtrommel mit äußerst stabilem Federrückzug, auf die ein hochflexibles Stahlseil einlagig gespult wird. Die Trommellagerung hat den Umfang von 150 mm und wird durch die Welle des angeflanschten Drehgebers mit potentiometrischem Gebersystem übernommen, das die Widerstandswerte von 0 bis 1 $k\Omega$ aufweist. Die Mess-

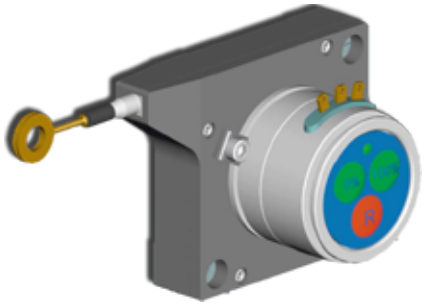


Abbildung 3.15: FSG - Seizugsensor

länge des Sensors ist auf 125 mm beschränkt. Sein Messseil hat den Durchmesser von 0,8 mm und ist am Ende mit einer Befestigungsscheibe bestückt.

Damit der Seilzugsensor seinen Zweck als Drehwinkelmesser erfüllen kann, wird er hinter der Kabine des Zugwagens fest angeschraubt. Wobei sein Messseil auf die Hälfte der Seillänge bzw. in der mittleren Position aufgespannt und unter dem Auflieger im Einstich einer dafür speziellgefertigten Rotationsscheibe befestigt wird. (sehe Bild 3.16). Bei der Linksdrehung des Aufliegers und somit der Scheibe zieht sich das Seil noch mehr aus dem Gehäuse raus und erzeugt damit die höhere Widerstand. Bei der Rechtsdrehung lockert sich das Seil, wodurch der Widerstand kleiner wird. Durch solche Widerstandsveränderungen werden vom Sensor analoge Signale erzeugt, die sofort vom Analog-Digitalkonverter des Mikrokontrollers aufgenommen und in digitale Werte umgewandelt werden. Aus der folgenden Tabelle, die die gemessenen digitalen Werte bei der unterschiedlichen Winkelposition des Aufliegers zu dem Zugwagen zeigt, wird es deutlich, dass die Winkelgenauigkeit etwa $\pm 2 - 3°$ beträgt.

-20°	-15°	-10°	-5°	0°	5°	10°	15°	20°
158	153	148	141	138	132	127	121	115
-22	-17	-12	-5	$\leftarrow 0 \rightarrow$	$+6$	$+11$	$+17$	$+23$

Tabelle 3.3: Werteveränderungen bei unterschiedlichen Drehwinkeln

Die Werte können bei jedem Einsatz des Fahrzeugs unterschiedlich variieren, da es vorkommen kann, dass der Seilverlängerung sich ausdehnt, oder dass bei der Initialisierung der Auflieger zu dem LKW-Zugwagen nicht genau auf die Nullposition ausgerichtet worden ist. Dabei werden die Werte genommen, die bei der Nullposition gemessen worden sind. In der Annahme, dass ein Wertunterschied einem Grad entspricht, lassen sich die Gradwinkel ω nach folgendem Prinzip berechnen: $\omega = Nullpos - Drehpos$ (Sehe die Tabelle 3.3)

Auffallend ist, dass der Anhänger bei einer Kurvenbewegung während der Vorwärtsfahrt einen maximalen Drehwinkel von $180°$ im Bereich von $-90°$ bis $+90°$ aufweist. Dies wird

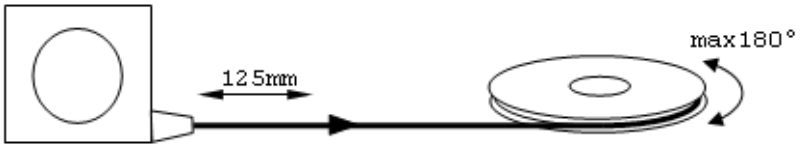


Abbildung 3.16: Seizugsensor - Drehscheibe-Verbindung

bei der Berechnung des Durchmessers der Drehscheibe berücksichtigt, indem der Umfang des Kreises zwei Mal so groß sein muss wie die Länge des Messseils des Sensors. $u = 2 * 125mm$ Nach folgender Formel lässt sich der Durchmesser d berechnen:

$$d = \frac{u}{\pi}$$

3.2.4 AT90CAN128-Bord

Um sowohl die Ansteuerung der Servo- und Gleichstrom-Motoren, als auch die Verarbeitung der analogen Ausgangsdaten der Sensoren des Fahrzeugs zu ermöglichen, wurde ein leistungsstarker AVR - Mikrokontroller der Firma ”ATMEL” eingesetzt.

Abbildung 3.17: Mikrokontroller - AT90CAN128

Es handelt sich dabei um einen AT90CAN128 Mikrokontroller, der auf der Abbildung 3.17 dargestellt ist, in einer 8 Bit RISC Architektur mit 133 Instruktionen und 32x8 GPW- und PC-Register. Der AT90CAN128 arbeitet mit einer Taktfrequenz von 8 bzw. 16MHz, von der übrigens seine Versorgungsspannung abhängig ist. Im 8MHz-Betriebsmodus muss er mit einer Spannung von 2.7 und im 16 MHz-Betriebsmodus mit einer Spannung von 4.7V versorgt werden. Da unser Roboter mit einer 12V Akkubatterie betrieben wird, wurde dazwischen ein

Spannungswandler geschaltet, welcher die nötige Spannung für den Mikrokontroller wandelt. Außerdem können bei dem Mikrokontroller 5 spezielle Modis aktiviert werden, ”Sleep-Modus” genannt, bei denen der Stromverbrauch vom aktuellen Arbeitszustand und der Prozessorbelastung abhängt. Als Speicher sind auf dem Mikrokontroller 128KBytes System-Flashspeicher, 4KBytes EEPROM und 4KBytes externes SRAM 5 vorhanden. Zudem kann es bis zu 64KBytes aufgerüstet werden. Für die Kommunikation, Programmierung und das Remote-Debugging gibt es eine Master/Slave-SPI-, RS-232- 5 und JTAG- 5 Schnittstellen. Dazu besitzt der AT90CAN128 einen CAN-Kontroller 5, der zu CAN-2A (mit 11Bit-Identifire) und CAN-2B (mit 29Bit-Identifire) kompatibel ist.

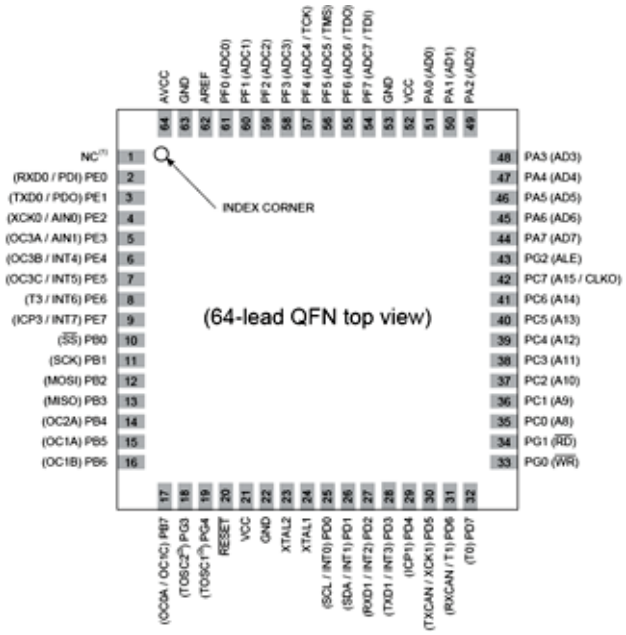


Abbildung 3.18: Pinbelegung - AT90CAN128

Der verwendete AT90CAN128 ist in einem TQFP 5 - Gehäuse mit 64 Pins angefertigt. 53 Pins werden als programmierbare Input/Output Anschlüsse verwendet. Mehrere davon können aber auch andere bestimmte Funktionen wie externe Interrupts, Speicheranbindung, AD -Konverter, Timer ausführen. Das Bild 3.18 zeigt die Pinbelegung des AT90CAN128. Der Mikrokontroller hat einen 8-bit synchronen, einen 8-bit asynchronen und zwei 16-bit synchronen integrierten Timer bzw. Zahler, die alle über einen 10-bit Prescaler verfügen. Zusätzlich ist der Mikrokontroller mit weiteren Funktionen wie programmierbares Watchdog-Timer zur Erkennung von fehlerhaften Funktionen und PWM - Puls-Breite-Modulation zur Generierung der Signale für Servo- und Gleichstrommotoren ausgestattet.

Alle diese Eigenschaften machen aus AT90CAN128 einen leistungsfähigen Mikrokontroller, der ideal für die privaten und industriellen Anwendungen geeignet ist.

Analog Digital Konverter

Der AT90CAN128 besitzt ein 10-Bit Analog-Digitalkonverter (ADC), das eine Konvertierung der von Sensoren erzeugten analogen Signale in digitale Werte erlaubt. Dabei reicht die 10-Bit Auflösung für die maximale dezimale Zahl 1024. Da die in Rahmen dieses Projekts genutzten Sensoren die Messwerte aufweisen, die die Dezimalwerte von 256 nicht überschreiten, wird die 8-Bit Auflösung völlig ausreichen. Zusätzlich verfügt der Mikrokontroller über insgesamt 8 Kanäle, die für ADC geeignet sind. (Sehe die Abbildung "Pinbelegung der AT90CAN128") Allerdings werden in Rahmen dieses Projekts nur die ersten 4 Kanäle für die Konvertierung genutzt, da die Anderen für JTAG- Funktionen reserviert sind.

Für Digitalisierung von analogen Sensorsignalen braucht der Mikrokontroller eine Referenzspannung, deren Höhe von dem maximalen Wert der konvertierten Spannung abhängt. Zur Auswahl gibt es interne 2,56V und externe Referenzspannungen, die durch einen zusätzlichen Kondensator an den Pin AREF angelegt wird. Für die Konvertierung der analogen Signale der Abstandsensoren und des Seilzugsensors wird in diesem Projekt die interne 2,56V Referenzspannung benötigt, die man aktiviert, indem man in dem ADC- Multiplexer Selection Register beide Pins (REFS1 und REFS0) auf HIGH setzt. Die ADC- Umwandlung resultiert sich aus der Formel:

$$ADC = \frac{V_{\mathrm{IN}} * 1023}{V_{\mathrm{REF}}}$$

Wo V_{IN} eine zu digitalisierende Spannung und V_{REF} eine Referenzspannung 2,56V sind. Nach der ADC- Umwandlung werden die Ergebnisse als eine Dezimalzahl in ADC- Dataregistern (ADCL, ADCH) abgespeichert.

Wenn die Messdaten von den drei Abstandsensoren und dem Seilzugsensor (Sehe Kapitel 3.2.3) vom Mikrokontroller erfasst worden sind, werden sie in ein 4 Byte großes Datenpaket gepackt und an CAN-Bus gesendet.

3.3 Architektur des Positionsbestimmungskonzepts

Das gesamte Konzept zur Positionsbestimmung erfordert synchrone Zusammenarbeit von mehreren Komponenten. Zum ersten ist das die Laserscanneranlage, die die Distanzmesswerte liefert, indem sie die momentane Umgebung rund um sich abtastet. Zum Zweiten sind das die Steuerung und Sensoren des im Kapitel 3.2.1 beschriebenen LKW-Modells. Und schließlich der Positionsbestimmungsalgorithmus, der auf die Messdaten von den ersten zwei Komponenten angewiesen ist. Der Arbeitsvorgang des gesamten Konzepts kann in zwei Phasen aufgeteilt werden: Initialisierung und Normal-Betrieb. Bei der Initialisierung

geht es in der ersten Linie darum, dass alle Komponente ihre Starteinstellungen einnehmen und entsprechende Funktionen durchführen. In der folgenden Tabelle 3.4 werden die vorerwähnten Funktionen der Initialisierung Reihenfolge nach aufgelistet.

LKW-Modell	Der Zugwagen und sein Auflieger müssen in einer Linie und 180°-Winkel bilden
LKW-Steuerung und Sensoren	Alle Start einstellungen des Mikrokontrollers CAN, ADC, Abstandsensoren, Seilzugsensor Lenkung, PWM usw.
Laserscanner	Erste Raumscannung, Messdaten in Textdatei speichern
Positions-bestimmungs-algorithmus	Nach Messwerten der ersten Raumscannung werden Polygon und Koordinatenursprung bzw. die Null-Position des LKW bestimmt

Tabelle 3.4: Funktionen der Komponenten bei der Initialisierung

Erst nachdem alle Initialisierungsprozesse beendet sind, startet der Normal-Betrieb des Konzepts. In dieser Arbeitsphase findet die eigentliche Positionsbestimmung statt. Zur Positionsbestimmung sind zwei unterschiedliche Modelle erarbeitet worden. Ein Modell basiert auf der kontinuierlichen Scannung der momentanen Umgebung während der Einsatzfahrt. Das andere Modell beruht auf der Scannung der Umgebung nach bestimmten Intervallen. Im Grunde der beiden Modelle liegt aber der gleiche Algorithmus zur Positionsbestimmung. Welche Unterschiede die Modelle in ihrer Struktur aufweisen, wird in nächsten Abschnitten dieses Kapitels erläutert.

Modell mit intervallen-Raumscannung

Bei dem Funktionsprinzip des Positionsbestimmungsmodells, das auf der Abbildung 3.19 dargestellt ist, geht es in der ersten Linie darum, dass bei einer Einsatzfahrt des autonomen mobilen Fahrzeugs die Abtastung des Raums mit dem Laserscanner und somit die Positionsbestimmung nach gewissen Zeitabständen oder nach einer bestimmten gefahrenen Strecke verlaufen soll. Dabei werden alle Veränderungen in der Geschwindigkeit und in der Lenkung beim autonomen mobilen Fahrzeug erfasst. Da das im Kapitel 3.2.1 beschriebene LKW-Modell für seine Fahrt spezielle Befehle von einem Steuerungsgerät annimmt, können diese Befehlsdaten für die Berechnung der Koordinaten der geschätzten Position benutzt werden, die als Parameter an das Positionsbestimmungsprogramm übergeben werden.

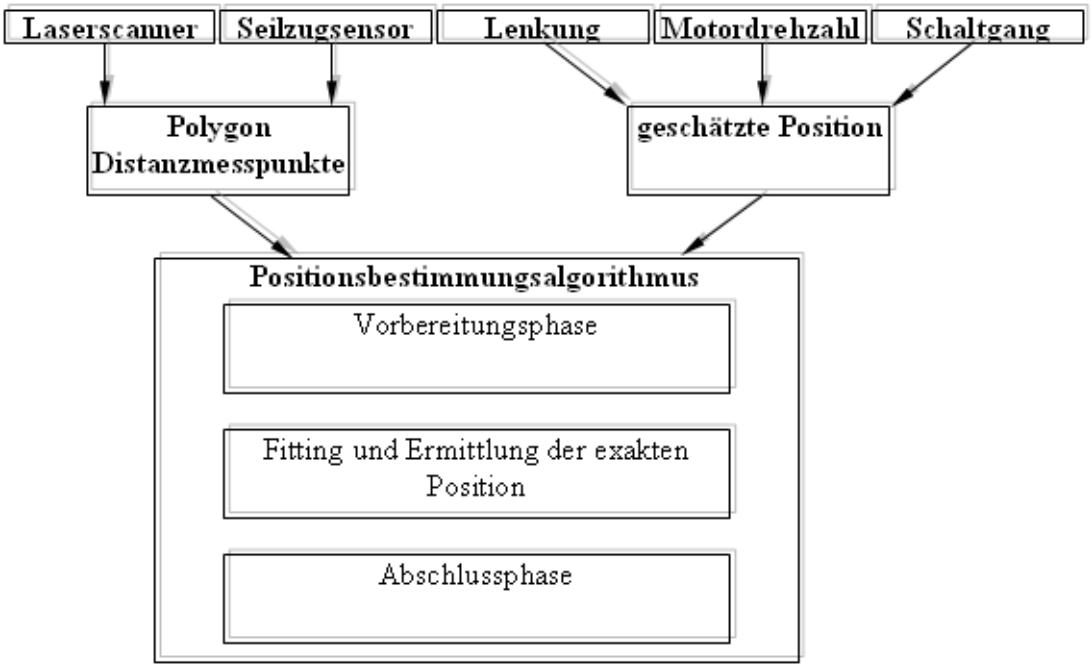


Abbildung 3.19: Modell-1 mit intervallen-Raumscannung

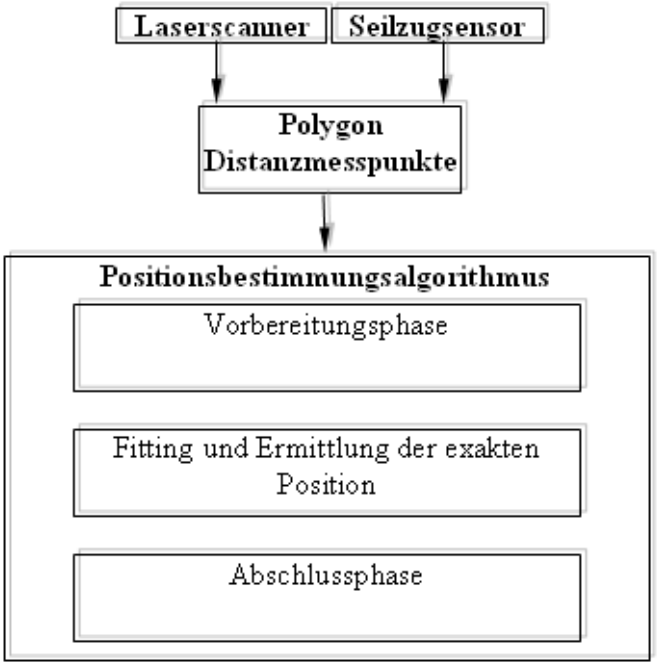


Abbildung 3.20: Modell-2 mit kontinuierlicher Raumscannung

Modell mit kontinuierlicher Raumscannung

Meistens müssen sich die autonomen mobilen Roboter in ihrem Einsatzgebiet sehr langsam und akkurat bewegen. Bei solchen Einsatzgebieten kann es möglicherweise um die gefährliche und nicht freie Landschaften handeln. In diesem Fall kann das Positionsbestimmungsmodell mit kontinuierlicher Abtastung der Gegend eingesetzt werden. (Sehe die Abbildung 3.20) Bei diesem Modell wird keine externe Bestimmung der geschätzten Position des autonomen Roboters durchgeführt. Deshalb ist das Gebrauch von Drehzahl- und Lenkungs-Sensoren irrelevant. Da aber im Grunde der beiden Modelle das gleiche Positionsbestimmungsverfahren liegt und da die geschätzte Position als einer der wichtigsten Parameter bei der Ermittlung der exakten Positionskoordinaten agiert, wird statt der externen geschätzten Positionsbestimmung eine interne geschätzte Positionsbestimmung durchgeführt. Die interne Bestimmung der geschätzten Positionskoordinaten läuft im Grunde genommen dadurch ab, dass die ermittelte Position bei dem nächsten Positionsbestimmungsablauf zu der geschätzten Position wird. Die einzige Voraussetzung ist, dass das autonome mobile Fahrzeug nur im Schneckentempo bis auf 3km/Std oder weniger als 1M/sek sich fortbewegen kann. Die Abgrenzung der Geschwindigkeit des autonomen Fahrzeugs hängt dabei von der Schnelligkeit des Positionsbestimmungsalgorithmuses und von der Größe des Fitting-Bereichs ab. Der Ablauf des ganzen Algorithmuses wird im nächsten Kapitel ” Realisierung des Positionsbestimmungsalgorithmus” beschrieben. Die Abbildung 3.21 stellt visuell einen möglichen Ablauf bei der internen Bestimmung der geschätzten Position dar.

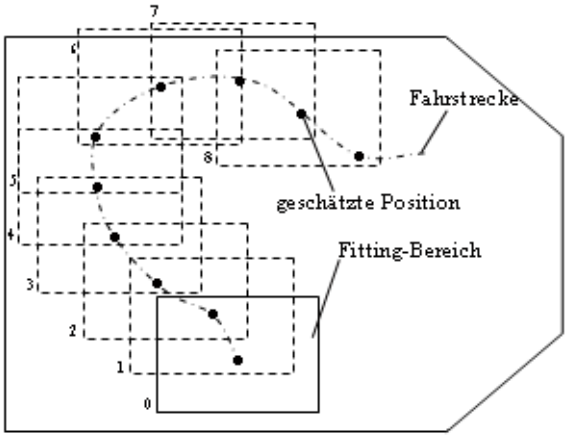


Abbildung 3.21: Ablauf bei der internen Berechnung der geschätzten Position

3.4 Realisierung des Positionsbestimmungskonzepts

3.4.1 Positionsbestimmungsalgorithmus

Der eigentliche Positionsbestimmungsalgorithmus besteht aus 3 Arbeitsphasen Vorbereitungsphase, Hauptphase, wo die Positionsberechnung stattfindet, und Abschlussphase.(Sehe das Bild 3.22) Bei der ersten Aktivierung des Positionsbestimmungsalgorithmus startet ein Initialisierungsvorgang, bei dem aus der ersten Raumscannung die Ursprungskoordinaten und die Ausrichtung des autonomen Roboters und das Polygon bestimmt werden.

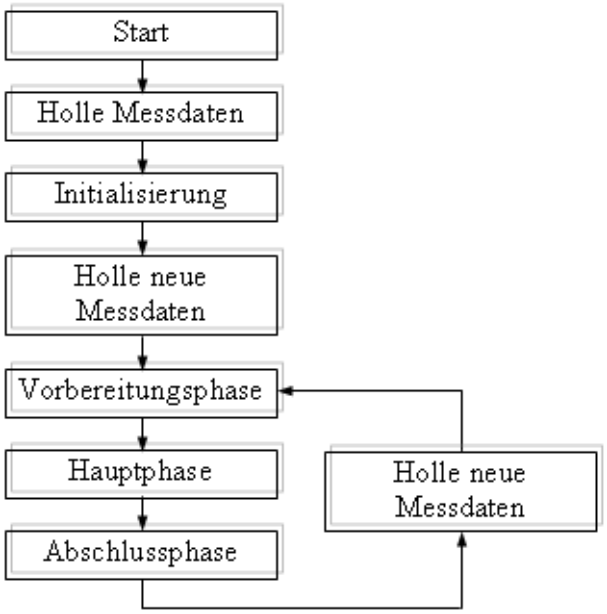


Abbildung 3.22: Ablauf des Positionsbestimmungsalgorithmus

In der ersten Vorbereitungsphase werden alle Distanzmesspunkte bzw. ihre Koordinaten bezüglich der vorderen Lenkachse nach dem Algorithmus, die im Kapitel 3.2.1 beschrieben wurde, konvertiert. Dabei werden die gemessenen Koordinatenwerte je nach dem Vorgang ”Drehen” oder ”Verschieben” gemäß im Kapitel 3.1.2 beschriebenen Verfahren in kartesische oder Polar-Koordinaten umgewandelt. Zusätzlich wird in dieser Phase die geschätzte Position berechnet. Die externe Berechnung der geschätzten Positionskoordinaten des eingesetzten LKW-Modells kann sein Mikrokontroller übernehmen. Es werden einfach die neuen geschätzten Koordinaten an den Hauptrechner versendet, wo sie zu der aus früherer Positionsberechnung ermittelten Koordinaten dazuaddiert werden. Wenn es aber um die interne Bestimmung der geschätzten Position geht, gilt hier die Regel, dass die ermittelte Position bei der nächsten Berechnung als geschätzte genommen wird.

Als nächstes wird die geschätzte Position zurück zu dem Ursprungspunkt versetzt. Die von der geschätzten Position aus gemessene Distanzmesspunkte werden ebenfalls durch ”Drehen” und ”Verschieben” bezüglich des Ursprungs konvertiert. Ab jetzt tritt die nächste Hauptphase der Positionsbestimmung vor.

In der Hauptphase des Algorithmus dreht es sich um die Berechnung des minimalen mittleren Abstandes der Distanzmesspunkte zum Polygon und um die Bestimmung der exakten Abweichungskoordinaten von der geschätzten Position des Fahrzeugs in der Umgebung. Dabei hängt die Genauigkeit und Schnelligkeit der Positionsberechnung von der Größe und Schrittauflösung des Fitting-Bereichs ab. (Sehe den Kapitel 3.1.3 und die Abbildung 3.2). Der Fitting-Bereich wird in drei Ebenen aufgeteilt: Richtungswinkel, $x-$ und $y-$ Koordinaten, deren Größen vom Programmierer oder dem Benutzer bestimmt werden können. Man kann es vorstellen wie eine 3D-Matrix, wo eine Stelle einer Auflösungseinheit entspricht. Genau in der Mitte dieser Matrix befindet sich der zu dem Ursprung versetzte Punkt der geschätzten Position. Deswegen ist es bei der internen Bestimmung der geschätzten Position von größerer Bedeutung, dass, bevor der nächste Ablauf des Fitting-Algorithmus stattfindet, der LKW-Modell mit geringerer Geschwindigkeit fährt und das Fitting-Bereich nicht verlässt.

Beim Ablauf des Algorithmus werden alle Stellen der Matrix und somit alle mögliche Koordinaten des Fitting-Bereichs durchgegangen. Und dabei wird für jede Stelle oder Koordinatenposition des Fitting-Bereichs die Berechnung des mittleren Abstandes gemacht. Die Abbildung 3.23) zeigt den Ablauf von dem Fitting-Algorithmus. Der Wert ”max” ist hier die maximale erlaubte Anzahl von Schritten für Richtungswinkel φ, $x-$ und $y-$ Koordinaten, die durch die Auflösung bestimmt werden. Die Auflösung kann dabei für jede diese Ebene einzeln gewählt werden.

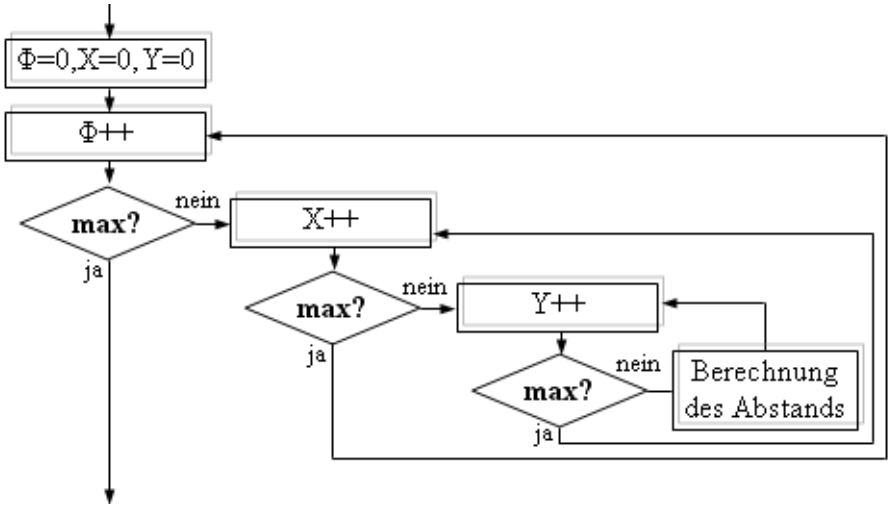


Abbildung 3.23: Ablauf des Fitting-Bereichs

Die Berechnung des mittleren Abstands zwischen den Distanzpunkten der geschätzten Position und dem Polygon spielt sich nach folgendem Prinzip ab:

- Die Berechnung des Abstands zum Polygon wird für jeden einzelnen Distanzpunkt zu jedem Polygonabschnitt durchgeführt.
- Als erstes sucht man den Abstandsvektor zwischen dem Distanzpunkt und dem Polygonabschnitteinheitsvektor.
- Danach wird das Skalarprodukt (Projektion auf den Polygonabschnitt) vom Polygonabschnitt und dem Abstandsvektor. Wenn der Skalar größer als der momentane Abschnitt oder kleiner 0 ist, wird ein anderer Abschnitt genommen.
- Wenn der Skalar im Größebereich des Abschnitts liegt, wird das Kreuzprodukt berechnet, um Flächeninhalt des von dem Abschnitts- und Abstands-Vektor aufgespannten Parallelogramms zu finden.
- Aus der Fläche des Parallelogramms wird die Höhe berechnet, deren Betrag dem gesuchten Abstand gleicht.
- Wenn alle Abstände für die Distanzpunkte berechnet und gespeichert wurden, wird aus ihrer Menge und Länge der Mittelwert ermittelt.
- Anschließend wird der gesuchte mittlere Abstand bezüglich der Koordinaten des Fitting-Bereichs gespeichert.

Nachdem alle Positionskoordinaten des Fitting-Bereichs durch sind, wird aus der Menge der mittleren Abstände der minimale Abstand gesucht. Die Stellenposition des minimalen Abstands entspricht der Positionskoordinaten in dem Fitting-Bereich und somit der Abweichung bei der geschätzten Position für Richtungswinkel φ, $x-$ und $y-$ Koordinaten.

In der Abschlussphase werden die geschätzten Positionskoordinaten nach den gefundenen Abweichungen korrigiert, indem abgeweichten Richtungswinkel φ, $x-$ und $y-$ Koordinaten zu den geschätzten Positionskoordinaten zusammen addiert werden. Damit wird die exakte Position des autonomen mobilen Fahrzeugs ermittelt.

3.5 Softwareimlementierung

3.5.1 Software und Werkzeuge

AVR-Studio

Das AVR-Studio ist eine professionelle Entwicklungsumgebung der Firma Atmel für die Programmierung der AVR-Mikrocontroller, deren aktuelle Version man kostenlos von der In-

ternetsite der Firma herunterladen kann. Die Programmierung kann sowohl in Assembler als auch in C erfolgen. Allerdings für die C-Programmierung muss bei der Installation des AVR-Studios der WIN-AVR C/C++ Compiler aus der GNU Compiler-Kollektion AVR-GCC für AVR-Mikrocontroller installiert werden. Das AVR- Studio-Softwarepaket besteht aus einer Projektverwaltung, einem Editor und einem Debugger, der mit dem integrierten Simulator, einem JTAG-Adapter oder einem In Circuit Emulator genutzt werden kann. Auf der Abbildung 3.24 wird das grafisches Interface des AVR- Studios dargestellt.

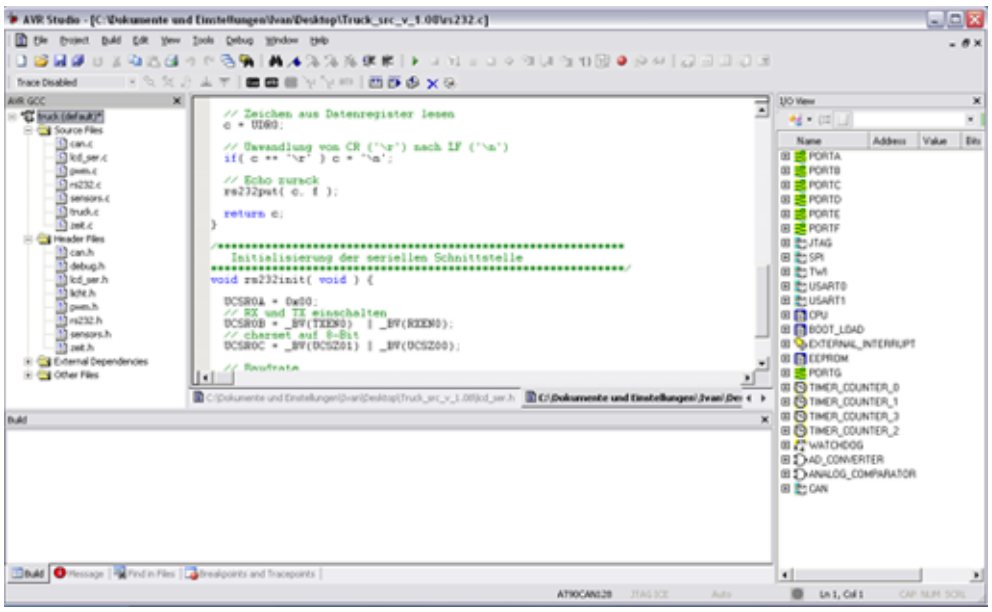


Abbildung 3.24: AVR-Studio

Eclipse

Eclipse ist ein universales Open-Source-Framework zur Entwicklung von Programmen aller Art. Im Vordergrund steht die Entwicklungsumgebung (IDE), die die Nachfolgerin von ”IBM Visual Age for Java 4.0” ist. Im Jahr 2001 wurde der Quellkode von Eclipse für weitere Entwicklung freigegeben. Primär wird Eclipse als Java-IDE verwendet, ermöglicht ebenfalls die Programmierung in mehreren anderen Programmier- und Scriptsprachen, wie C/C++, Perl, PHP, HTML, XML, grafisches UML usw. Dafür müssen nur die entsprechenden Tools nachinstalliert und konfiguriert werden. Es wird zum Beispiel für die Unterstützung von C/C++ Programmierung das C/C++ Development Tooling benötigt, das den GNU GCC Compiler und Bibliotheken beinhaltet. Die Entwicklungsumgebung von Eclipse stellt die folgende Abbildung 3.25 dar.

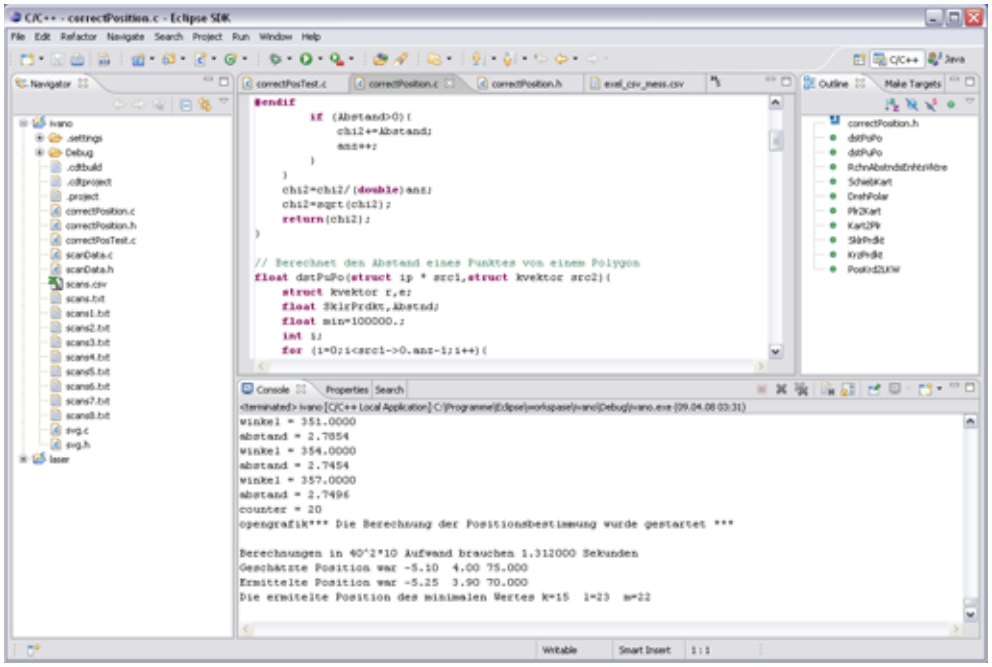


Abbildung 3.25: Eclipse

SVG

Zur visuellen Darstellung der zweidimensionalen Konturenabbildungen der Einsatzumgebung des mobilen Roboters hat sich Scalable Vector Graphics - SVG am effektivsten erwiesen. SVG ist ein XML basiertes Dateiformat, das von meistverbreiteten Webbrowsern unterstützt wird. Mit SVG lässt sich die Vektor-, Raster- und Animations-Grafik darstellen. SVG-Dateien sind leicht zu implementieren und können mit einfachem Texteditor bearbeitet werden. Es gibt jedoch für die Bearbeitung von SVG spezielle Programme mit integrierten Editor und Viewer, zum Beispiel Inkscape, das man frei aus dem Internet runterladen kann.

3.5.2 Implementation der Software für den AT90CAN128-Mikrocontroller

Zur externen Bestimmung der geschätzten Position des mobilen Fahrzeugs soll es nach folgenden Kriterien vorgegangen werden.

- Alle per WLAN empfangene Nachrichten werden vom Mikrokontroller erst abgespeichert

- Die abgespeicherte Daten werden nacheinander bearbeitet und somit die nächste Kursfahrt des mobilen Fahrzeugs bestimmt
- - Während der Kursfahrt werden aus den eingesetzten Steuerungsdaten und aus der Kursfahrzeit die Koordinaten der geschätzten Position berechnet und abgespeichert
- Die geschätzten Koordinaten werden nach jede Kursfahrt an die Positionsbestimmungsalgorithmus gesendet

Um dieses Arbeitsschema zu verwirklichen müssen mehrere Funktionen des Mikrokontrollers implementieren werden. Die Tabelle 3.5 gibt die Zusammenfassung aller benötigten Funktionen zur Bestimmung der geschätzten Positionskoordinaten an.

Funktionen	Beschreibung
void canInit();	Initialisierung des CAN-Busses
int canReceive(int *data, uint1 *id);	Empfangen der Daten vom CAN-Bus
void canSend(char *buf, uint laenge, uint id);	Senden der Daten an CAN-Bus
void pwmInit();	Initialisierung des PWM-Timers
void setMotor(int speed);	Setzen die Geschwindigkeit
void setLenkung(int lenkung);	Setzen den Lenkwinkel
void setGetriebe(int gang);	Setzen den Schaltgang
void sensorsInit();	Initialisierung des Digital-Analog-Konverters und T3 Timers
void readSensor(uint sensorNum);	Auslesen von Sensorenmesswerten
void newRoute(uint newCnt, int newLenkung, int newSpeed, int *x, int *y);	Bestimmen der neuen Strecke und X-, Y- Koordinaten
void timerInit();	Initialisierung des Timers
void rs232Init();	Initialisierung der RS232-Schnittstelle.
int rs232put(char c, FILE *f);	Ausgabe eines Zeichens auf die RS232-Schnittstelle.
int rs232get(FILE *f);	ILesen eines Zeichens von der RS232-Schnittstelle.

Tabelle 3.5: Zusammenfassung implementierter Funktionen für AT90CAN128

Die Hauptfunktionen für die Berechnung der geschätzten Positionskoordinaten sind `canReceive`,`newRoute`und `canSend`.

`canReceive`: mit dieser Funktion werden die Daten zur Steuerung des Fahrzeugs empfangen. Gesteuert werden die Geschwindigkeit des Fahrzeugs, der Lenkwinkel seiner vorderen Achse und die Schaltgang.

`newRoute`: Funktion setzt die empfangenen Parameter für Regulierung der Motoren ein. Für die Fahrzeit, nachdem die Motoren neue Befehle erhalten haben, sorgt der Timer. Parallel dazu werden die geschätzten $X-$ und $Y-$ Koordinaten berechnet. Die Technik zu Berechnung der geschätzten Position wurde im Kapitel 3.1.4 beschrieben.

`canSend`: mit der Funktion werden die Sensorenmesswerte sowie die $X-$ und $Y-$ Koordinaten der geschätzten Position als Datenpakete an CAN-Bus übergeben. Nachdem die Daten an CAN-Bus gesendet wurden, werden die $X-$ und $Y-$ Koordinaten auf 0 gesetzt. Die Datenpakete von CAN-Bus werden dann weiter per WLAN an den Hauptrechner bzw. zur Positionsbestimmung gesendet.

3.5.3 Implementierung der Positionsbestimmungsalgorithmus

Die Implementierung des Programms für die Positionsbestimmung wurden komplett in der Programmiersprache C geschrieben. Die aufgebrachte Tabelle 3.6 gibt eine Zusammenfassung aller für die Positionsbestimmung benötigten Funktionen an. In folgenden Abschnitten dieses Kapitels werden die aufgelisteten Funktionen genauer beschrieben.

Messdaten

Die gescannten vom Laserscanner mit dem Winkelauflösung von $1\circ$ Daten werden in einer Textdatei als Polarkoordinaten paketweise gespeichert. Bevor die Daten zur Positionsbestimmung verwendet werden, müssen sie zuerst aus der Datei gelesen werden. Die Funktion `FILE* sdOpen(char*fname)` sucht in dem eingegebenen Pfad `fname` nach die Datei und liefert die Struktur- FILE mit gesamten Inhalt zurück. Als nächstes werden die benötigten Messdaten mit der Funktion `void sdPrepar(struct app* src, FILE* csv, int resNum, int paketNum);` ausgelesen und als Polarkoordinaten in die als Parameter übergebene Struktur `struc app*` gespeichert. Bei der Auswahl der Messdaten können die Drehwinkelauflösung mit dem Eingangsparameter `resNum` und die Anzahl der Datenpaketen, die für die Fehlerkorrektur nötig sind, mit `paketNum` bestimmt werden. Zur Fehlerkorrektur werden aus der eingegebenen Anzahle der Datenpakete die gleichen Distanzmesswerte ausgesucht, von denen der Mittelwert berechnet wird.

Messdatenkorrektur

Bevor die Messdaten für die Positionsbestimmung bereit sind, müssen sie bezüglich des LKW-Lenklachse korrigiert werden. Dafür gibt es die Funktion `void PosKrd2LKW(struct app* src, struct app* dst, float dphi);` .

Im Grunde genommen werden hier die Koordinaten der $y-$Achse entlang verschoben und nach eingegebenen `dphi` Winkel gedreht. Die y Werte sind die LKW-Abstände, die wie folgt deklariert sind:

```
# define DIST_LKW 440
# define DIST_AHG 80
```

Daten werden erst nach `DISTAHG` verschoben, dann nach `dphi` gedreht und schließlich nach `DISTLKW` noch mal verschoben.

Bei der Verschiebung müssen die Polarkoordinaten erst in die Kartesische mit der Funktion `void Plr2Kart (struct app* src, struct akp* dst);` umgewandelt werden. Die eigentliche Verschiebung passiert in der Funktion `void SchiebKart(struct akp* src, struct akp* dst,double dx, double dy);`, wo die Referenzen `dx` und `dy` zu den momentanen Koordinaten dazu addiert werden. Die Funktion `void RotatePolar(struct app* src, struct app* dst, float dphi);` steht für die Drehung der Koordinaten und um ihren Ursprung. Zuerst aber müssen die Koordinaten mit der Funktion `void Kart2Plr(struct akp* src, struct app* dst);` zurück in die Polare versetzt werden. Die Technik zur Konvertierung der Koordinaten aus kartesischen in die Polare und in der umgekehrter Richtung sowie die Koordinaten-Drehung und -Verschiebung wurde im Kapitel 3.1.2 beschrieben.

Da es mit zwei verschiedenen Koordinatensystemen gearbeitet wird, sind zum Speichern der Distanzmesswerte als vektorielle Größen folgende Datenstrukturen deklariert.

```
// Vektor in kartesische Koordinaten
struct kvektor {float x; float y;};
// Vektor in Polarkoordinaten
struct pvektor {float r; float phi;};

// alle Ortsvektoren in kartesische Koordinaten
struct akp { int anz;
             struct kvektor p[200];};
// alle Ortsvektoren in Polarkoordinaten
struct app { int anz;
             struct pvektor p[200];};
```

Polygondefinition

Nach der ersten Scannung des Raums muss man ein Polygon definieren, indem alle Messpunkte miteinander verbunden werden. Die dabei entstehenden Abschnitte werden zu Polygonkanten. Die Polygonwerte werden in einer speziell deklarierten Datenstruktur gespeichert:

```
struct ip { struct akp O;     // Ortsvektoren
     struct kvektor v[200];   // Abstandsvektoren
     struct kvektor ev[200]; // Einheitsvektoren von v
     float bv[200];           //  Betrag von v
};
```

Zuerst aber müssen dieser Struktur die Ortsvektorn übergeben werden. Danach werden die Abstandsvektoren, ihre Beträge und Einheitsvektoren des Polygons in der Funktion `void DefPolygon( struct ip * src);` berechnet. Das Berechnungsverfahren wurde im Kapitel 3.1.5 detailliert beschrieben.

Fitting-Bereich

Nachdem das Polygon definiert ist und alle Initialisierungsfunktionen beendet sind, beginnt der Normal-Betrieb der Positionsbestimmung. Der Hauptablauf des Positionsbestimmungsalgorithmus passiert in den folgenden drei `for`-Schleifen. In der ersten `for`-Schleife werden Größe und die Schrittauflösung für φ -Fitting-Bereich bestimmt. In diesem Fall zeigt der maximale Wert von `k` die Anzahl der Schritten von der unteren -10$\circ$ bis zu oberen +10$\circ$ Grenze des Fitting-Bereichs. Da es um den φ -Winkel handelt, wird nach jeder `k`-Inkrementierung eine Drehung `-10+k` aller Distanzpunktkoordinaten gemacht. In weiteren zwei `for`-Schleifen werden x- und y- Fitting-Bereiche bearbeitet. Dementsprechend müssen die gedrehten Koordinaten aus der ersten `for`-Schleife erst in kartesische konvertiert werden. Die Grenzen bei den x- und y- Fitting-Bereichen werden hier von -1m bis +1m bestimmt. Die Größe des Fitting-Bereich für x und y ist damit 2x2m und die Auflosung bei maximalen `l` und `m` von 40 ist 5cm. Die Größen der Fitting-Bereichen von φ, x und y können natürlich beliebig gewählt werden.

```
for (k=0;k<20;k++){
      RotatePolar(aPol, bPol, -10+k);
      Plr2Kart(bPol, aKrt);
      for (l=0;l<40;l++){
            MovieKart(aKrt, bKrt, -1+l*0.05, 0.0);
```

```
                for (m=0;m<40;m++){
                      MovieKart(bKrt, aKrt, 0.0, -1+l*0.05);
                      // Abtandberechnung
                }
        }
}
```

Berechnung des minimalen mittleren Abstands

Nach jedem Schritt im gesamten Fitting-Bereich wird die Abstandberechnung zwischen allen Distanzpunkten und dem Polygon durchgeführt. Dafür werden in die Funktion `double dstPoPo(struct ip * src1, struct akp * src2);` Strukturen des Polygons und der Distanzmesspunkten als Parameter übergeben. Diese Funktion ruft zu jeder Distanzmesskoordinate eine weitere Funktion `float dstPuPo( struct ip * src1, struct kvektor src2);` auf, wo der minimale Abstand von dem Punkt zu allen Polygonabschnitten in folgender Reihenfolge berechnet wird.

- Abstandsvektoren $\vec{a}_i$ von Polygonpunkten $\vec{p}_i$ zu dem Distanzpunkt $\vec{d}$ nach der Formel $\vec{a}_i = \vec{d} - \vec{p}_i$
- Skalarprodukt bzw. Projektion des Abstandsvektors $\vec{a}_i$ auf Polygonabschnitt $\vec{p}_i$. Die Funktion dafür ist `float SklrPrdkt(struct kvektor a, struct kvektor b);`
- Kreuzprodukt bzw. Fläche des Parallelogramms mit `float KrzPrdkt(struct kvektor a, struct kvektor b);` bestimmen. Den Abstand berechnen, in dem die Fläche durch die Länge des Polygonabschnitts geteilt wird.
- Minimalen Abstand suchen und zurück an `dstPoPo` liefern.

Aus allen von `dstPuPo` berechneten Abständen wird ein Mittelwert gebildet, das dann in einem dreidimensionalen Array, dessen Länge dem Fitting-Bereich entspricht, unter gleichen `k, l, m` Indexen gespeichert wird. Wenn Berechnungen für alle möglichen Kombinationen des Fitting-Bereichs beendet sind, wird aus gespeicherten mittleren Abständen der minimale Wert ermittelt. Die Indexkombination von `k, l, m` zeigt dann die Abweichung in den Koordinaten der geschätzten Position von der richtigen Position. Es bleibt nur noch eine Korrektur der geschätzten Position durchzuführen.

Funktionen	Beschreibung
FILE* sdOpen(char*fname);	Öfnen der Textdatei mit Messdaten vom Lasescanner
void sdPrepar(struct app* src, FILE* csv, int resNum, int paketNum);	Zusammenstellung der Messdaten nach Drehwinkelauflösung und Fehlerkorrektur durch Berechnung des Mittelwertes
void Plr2Kart(struct app * src, struct akp *dst);	Konvertierung der Polarkoordinaten der Messpunkten in die kartesische Koordinaten
void Kart2Plr(struct akp * src, struct app *dst);	Konvertierung der kartesischen Koordinaten der Messpunkten in die Polarkoordinaten
void MoveKart(struct akp * src, struct akp *dst, double dx, double dy);	Verschiebung der Koordinaten nach $\Delta x-$ und $\Delta y-$ Referenzkoordinaten
void RotatePolar(struct app * src, struct app *dst, float dphi);	Drehung der Koordinaten nach dem $\Delta\varphi$ Referenzwinkel
void PosKrd2LKW(struct app *src, struct app *dst, float dphi);	Konvertierung der Dinstanzmesspunkten bezüglich der vorderer LKW-Lenkachse
void DefPolygon(struct ip * src);	Bestimmung des Polygons
double dstPoPo(struct ip * src1, struct akp * src2);	Bestimmungs des mitleren Abstandes zwischen Dinstanzmesspunkten und dem Polygon
float dstPuPo(struct ip * src1, struct kvektor src2);	Berechnug des Abstandes zwischen einem Punkt und dem Polygonabschnitt
float SklrPrdkt(struct kvektor a, struct kvektor b);	Berechnung des Skalarprodukts zweier Vektoren
float KrzPrdkt(struct kvektor a, struct kvektor b);	Berechnung des Kreuzprodukts zweier Vektoren

Tabelle 3.6: Funktionen des Positionsbestimmungsalgorithmus

4 Evaluierung

4.1 Testierung der Positionsbestimmungsanwendung

Bei der Entwicklung des Positionsbestimmungsalgorithmus wurden mehrere Tests mit unterschiedlichen Messdaten durchgeführt.

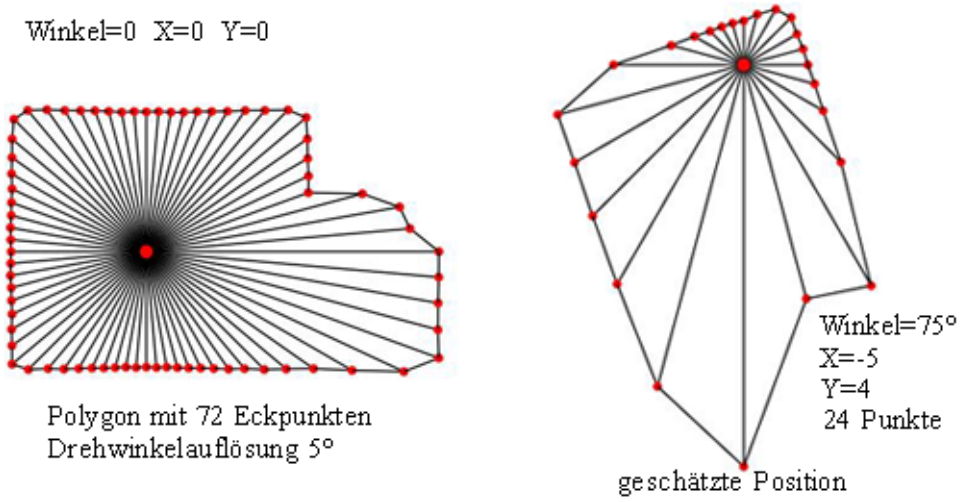


Abbildung 4.1: Polygon und Distanzpunkte von einer Papierskizze

In dem Frühstadium der Entwicklung wurden auf Papier gezeichnete Polygone verwendet. Der Ursprung bzw. die Startposition des Fahrzeugs und die geschätzte Position wurden zufällig ausgewählt. Dabei wurden die Distanzpunkte und Polygoneckpunkte in Maßstab 1cm:1m mit einem Lineal ausgemessen. Bei der Polygondefinition ist die Winkelauflösung auf $5°$ begrenzt worden, was bei einer Rundmessung 72 Messwerten entspricht. Für Bestimmung der Distanzpunkte wurde der Winkel von $15°$ mit 24 Messwerten bei der Rundscannung genommen. Die Abbildung 4.1zeigt das Polygon und die von der geschätzten Position aus gemessene Distanzpunkte. Obwohl die Abmessung der Distanzpunkte mit dem Lineal sehr präzise Messwerte erlaubt, wurden verschiedene Größen der Fitting-Bereiche mit unterschiedlichen Schrittauflösungen verwendet, um die Geschwindigkeit und Genauigkeit des Positionsbestimmungsalgorithmus zu testen. Die Tabelle 4.1 gibt die Testergebnisse der

Positionsbestimmung für ein Polygon mit 72 Eckpunkten und 24 Distanzpunkten bei unterschiedlichen Größen des Fitting-Bereichs an.

φ, x, y **Längen des Fitting-Bereichs in** $Grad$, m, m	**Schrittauflösung des Fitting-Bereichs für** φ, x, y	**erreichte Genauigkeit in** $Grad$ **und** cm	**erreichte Schnelligkeit in** sec	**Ermittelte Position in** φ, x, y
20°, 2, 2	20, 40, 40	1°, 5	1.302	φ=70° x=-5.25m y=3.90m
20°, 2, 2	40, 60, 60	0.5°, 3.3	5.838	φ=70.5° x=-5.22m y=3.84m
20°, 2, 2	40, 40, 40	1°, 5	2.563	φ=70.5° x=-5.25m y=3.85m
40°, 1, 1	40, 40, 40	1°, 2.5	2.624	φ=70° x=-5.22m y=3.82m
20°, 1, 1	20, 40, 40	1°, 2.5	1.321	φ=70° x=-5.25m y=3.90m
20°, 1, 1	40, 60, 60	0.5°, 1.6	5.818	φ=70.5° x=-5.23m y=3.83m

Tabelle 4.1: Testierung für geschätzte Positionskoordinaten φ=75°, x=-5m, y=4m mit unterschiedlichen Fitting-Bereichen

Als Hauptrechner für Testierung des Positionsbestimmungsalgorithmus fundierte ein handelsübliches Notebook mit dem ”Intel Pentium M ($1600\ MHz$)” Prozessor, IDE-Festplatte ($80\ GB$) und physikalischem Speicher von $1024\ MB$.

Die Tests haben gezeigt, dass die Positionsgenauigkeit sowohl bei Verkleinerung der Größen des Fitting-Bereichs als auch bei der Erhöhung der Schrittauflösung steigt. Die Geschwindigkeit des Algorithmus steigt jedoch nur dann, wenn die Schrittauflösung geringer wird.

Im nächsten Schritt wurden für den Positionsbestimmungsalgorithmus die Messwerte von dem im Kapitel 3.2.2 beschriebenen Laserscanner eingesetzt. Bei dem Versuch aus den Messdaten des Laserscanners eine visuelle Darstellung des gescannten Raums zu erstellen, sind einige Regelmäßigkeiten auffällig geworden. Die Illustrationen im Anhang zeigen die optischen Umrisse des gleichen Raums bei unterschiedlichen Drehwinkeln und Positionsverschiebungen des Laserscanners. Auf der Abbildungen A.1 A.2 A.3in der linken Spalte

werden die normalen erwarteten Konturen des Raums, die aus den Messdaten mit einer Drehwinkelauflösung von 3° gemacht wurden, und in der rechten Spalte sind die zaunmäßige Konturen dargestellt, die aus den Messdaten mit einer Drehwinkelauflösung von 2° gemacht wurden. Weitere Versuche mit dem gleichen Laserscanner ergaben das gleiche Ergebnis.

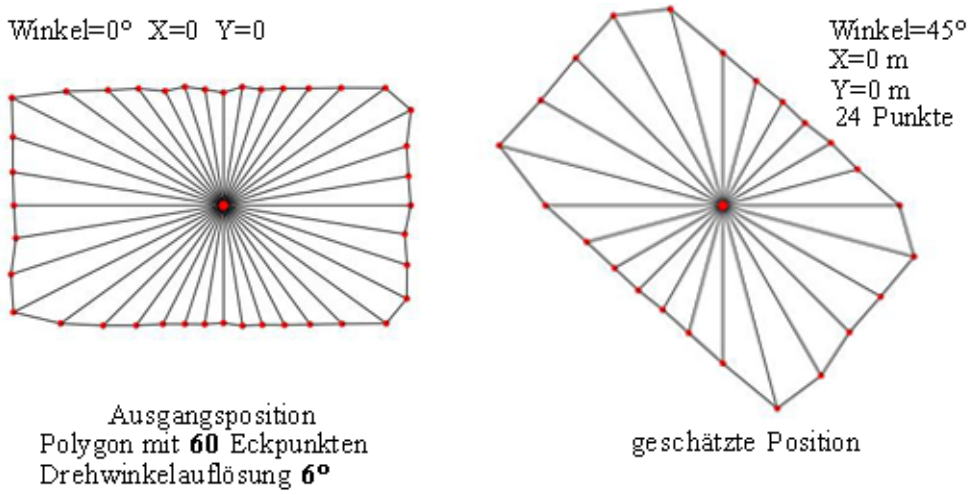


Abbildung 4.2: Polygon und Distanzpunkte aus Messdaten des Laserscanners

Aus diesem Grund werden die Drehwinkelauflösungen speziell so gewählt, dass sie durch 3 teilbar sind, damit für die Berechnung möglichst geradlinige Raumkonturen verwendet werden. Wie die Abbildung 4.2 zeigt, wird die Winkelauflösung für die Polygondefinition mit 6° und für die Distanzpunkten der geschätzten Position mit 15° genommen.

Für die Tests wurden mehrere Messungen mit dem zur Verfügung stehenden Laserscanner von verschiedenen Positionen gemacht. Nach jedem Positionswechsel des Laserscanners wurden die Änderungen der Koordinaten mit dem Lineal gemessen. Bei der Positionsbestimmung werden diese Positionskoordinaten als geschätzte Position eingesetzt. Nachdem alle benötigten Messwerte für den Positionsbestimmungsalgorithmus eingegeben wurden, hat das Programm folgende Ausgaben der ermittelten Position gemacht:

Position nicht geändert, nur gedreht `Geschätzte Position war` φ`=45.0°,` x`=0.00`m`,` y`=0.00`m `Ermittelte Position ist` φ`=40.0°,` x`=0.08`m`,` y`=0.00`m`Die Berechnung ist abgeschlossen in 0.981 sek`

Verschoben auf X-Achse, nicht gedreht `Geschätzte Position war` φ`=0.0°,` x`=-0.50`m`,` y`=0.00`m `Ermittelte Position ist` φ`=-7.0°,` x`=-0.57`m`,` y`=-0.08`m`Die Berechnung ist abgeschlossen in 0.993 sek`

Verschoben auf Y-Achse, nicht gedreht Geschätzte Position war φ=0.0°, x=0.00m, y=−0.40m Ermittelte Position ist φ=−7.0°, x=0.03m, y=−0.38mDie Berechnung ist abgeschlossen in 1.002 sek

Verschoben auf X- und Y-Achsen, nicht gedreht Geschätzte Position war φ=0.0°, x=−0.50m, y=−0.50m Ermittelte Position ist φ=−6.0°, x=−0.61m, y=−0.38m Die Berechnung ist abgeschlossen in 1.012 sek

Verschoben auf X- und Y-Achse, auch gedreht Geschätzte Position war φ=180.0°, x=−0.50m, y=−0.50m Ermittelte Position ist φ=177.0°, x=−0.57m, y=−0.38mDie Berechnung ist abgeschlossen in 1.093 sek

Grafisch lässt sich die ermittlelte Position in dem Fitting-Bereich wie folgt Darstellen 4.3. Diese Grafik entspricht der Situation **Position nicht geändert, nur gedreht**. Der Nullpunkt ist hier die geschätzte Position.

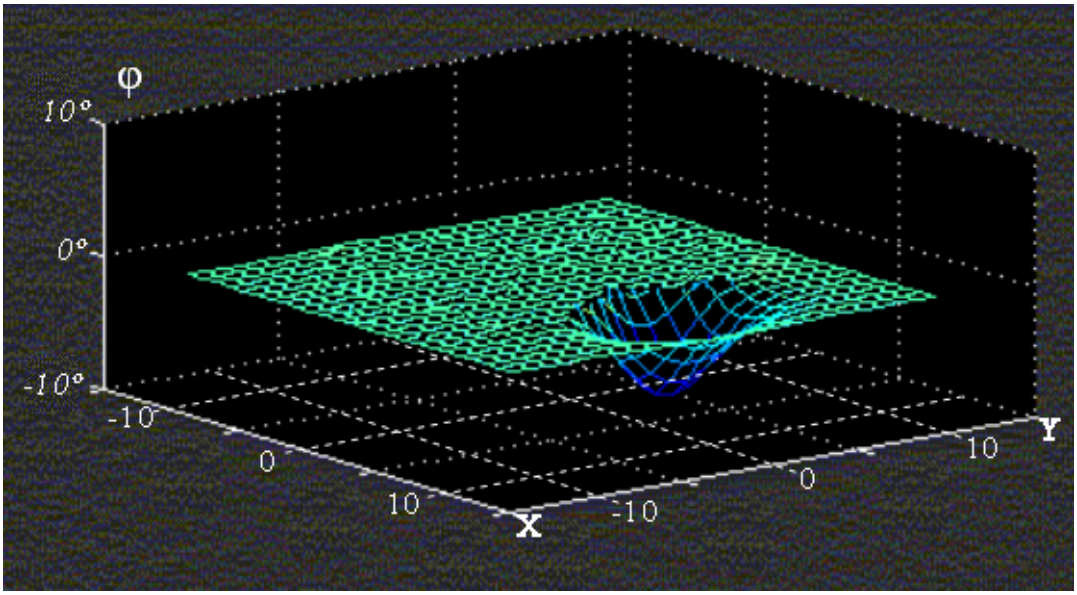


Abbildung 4.3: Grafische Darstellung der ermittelten Position im Fitting-Bereich

4.2 Mögliche Optimierungsvarianten

4.2.1 Kollinearität der Polygoneckpunkte

Oft ist eine Einsatzumgebung für den autonomen mobilen Roboter ein Raum, das mit vier oder mehreren geraden Wänden abgegrenzt ist. Wenn dabei eine Raumscannung mit einem Laserscanner gemacht wird, werden mehrere Strahlen auf eine Wand treffen und somit werden die gemessenen Distanzpunkte auf einer Gerade liegen. Genau in diesem Fall kann das Modell der kollinearen Punkte eingesetzt werden. Bei diesem Modell handelt es sich darum, dass während der Polygondefinition die kollineare Polygoneckpunkte bzw. die Punkte, die auf einer Linie liegen, aufgesucht werden. Durch diese Punkte wird dann eine Linie durchgezogen, die zu einer Kante des Polygons wird. Im Algemeinen heißt es, dass mehrere Polygonkanten zu einer Kante werden, damit das Polygon mit wenigen Kanten implementiert wird. Die Abbildung 4.4 zeigt eine mögliche Situation mit kollinearen Punkten.

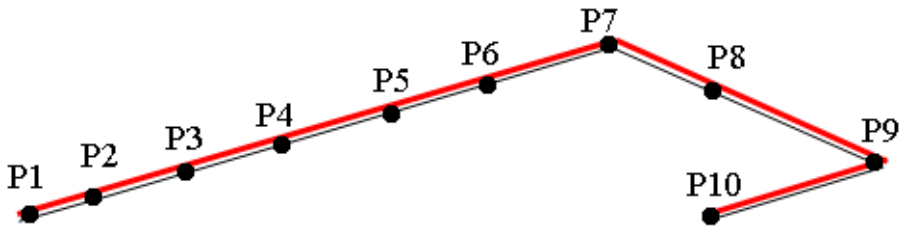


Abbildung 4.4: Kolineare Eckpunkte des Polygons

Für die Positionsbestimmung bedeutet das Polygon mit wenig Kanten eine schnellere Abarbeitung der Berechnungen der minimalen Abstände zu den Distanzpunkten. Damit dieses Model sich bewertet, müssen bei der Kantenberechnung mehr als drei kollineare Punkte geben. Denn, wenn es sich um eine runde Wand handelt, wird die Berechnung zu aufwendig. Da in Rahmen dieser Arbeit wurde die Positionsbestimmung auf die unbekannten Umgebungen gerichtet, bleibt dieses Optimierungsverfahren vorläufig nur in der Theorie.

4.2.2 Bewegung nach Bezierkurven

Die Verwendung der Bezierkurven ermöglicht eine glatte Fahrt zwischen drei oder mehreren Zielpunkten. (Sehe die Abbildung 4.5) Dabei wird die ganze Strecke in mehreren kleineren Abschnitten geteilt, damit das Fahrzeug in jeder Etappe seiner Fahrt die Positionskoordinaten leicht berechnen kann. Die Bezierkurventechnik wird in der parallelgeführten Diplomarbeit von Herrn Asche unter die Lupe genommen. Für die Positionsbestimmung bringt der Einsatz

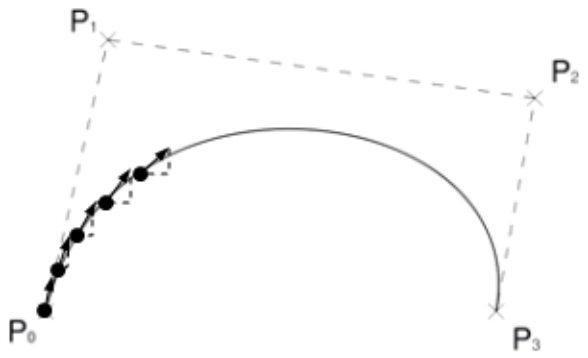


Abbildung 4.5: Bezierkurve

dieses Modells mehr Genauigkeit und Schnelligkeit. Denn Bezierkurventechnik bietet bessere Bestimmung der geschätzten Position. Und dadurch, dass die geschätzte Position wenig von der exakten Position abweicht, können bei dem Positionsbestimmungsalgorithmus die kleinere Fitting-Bereiche genommen werden.

5 Schluss

Die autonomen mobilen Systeme müssen oftmals ihre Aufgaben in einer völlig unbekannten Umgebung erledigen. Sie müssen in der Lage sein, ohne jegliche Hilfe von Außen, die Hindernisse zu erkennen, um sich kollisionsfrei bewegen zu können. Daher ist ganz wichtig, dass die autonomen mobilen Systeme sich schnell und sicher in der Einsatzumgebung orientieren können. Das Ziel dieser Bachelorarbeit bestand darin, aus den Messdaten der Sensoren der Laserscannanlage eines Roboters die exakte Position zu ermitteln.

Bei der Entwicklung des Positionsbestimmungskonzepts wurden zuerst andere konventionelle Positionsbestimmungsmöglichkeiten untersucht und theoretische Hintergrunde zur Positionsbestimmung in dem lokalen Konzept wie zum Beispiel geschlossene und abgegrenzte Räume erforscht. Anhand diesen Erkenntnisse wurden im Rahmen dieser Arbeit notwendige Komponenten zur Positionsbestimmung identifiziert, wie Hardware, Werkzeuge zur Software und mathematische Formeln, und ein auf einfachen mathematischen Mitteln basierender Positionsbestimmungsalgorithmus entwickelt, der dann in ein C Programm implementiert wurde. Weiterhin wurde die erstellte Anwendung zuerst mit auf dem Papier gezeichneten Messdaten und später mit echten gescannten Daten getestet.

Die Versuche die Positionsberechnung des Algorithmus zu beschleunigen, ergaben keine schlechten Zeitdifferenzen, allerdings mit kleineren Verlusten an der Genauigkeit. Für das entwickelte Konzept zur Positionsbestimmung wurden weitere Möglichkeiten zur Optimierung gezeigt und aufgezählt. Nichtsdestotrotz kann der entwickelte Positionsbestimmungsalgorithmus, obwohl es noch kleinere Mangelhaften gibt, schon heute eingesetzt werden.

Verwendete Abkürzungen

RFIG	Radio Frequency Identification
GSM	Global System for Mobile communications
UMTS	Universal Mobile Telecommunications System
CAN	Controller Area Network
RS232, 422	Recommended Standard 232, 422
EEPROM	Electrically Erasable Programmable Read-Only Memory
SRAM	Static Random Access Memory
TQFP	Thin Quad Flat Pack
JTAG	Joint Test Action Group

Literaturverzeichnis

[ABEL 2004] ABEL, Heinrich: *Global Positioning System Funktionsweise und mathematische Grundlagen*. 2004. – URL http://www2.hs-esslingen.de/~abel/gps/Abel-GPS.htm

[AG 2008] AG, SICK: *Lasersystem LD-OEM*. 2008. – URL www.sick.com

[FSG-Fernsteuergeräte 2008] FSG-FERNSTEUERGERÄTE: *Seillängenaufnehmer*. 2008. – URL www.fernsteuergeräte.de

[Hnilica 2006] HNILICA, Erich: *mathe-lexikon.at*. 2006. – URL http://www.mathe-lexikon.at

[HU-Berlin 2008] HU-BERLIN, Informatiker: *MagicMap*. 2008. – URL http://www2.informatik.hu-berlin.de/rok/MagicMap/index.htm

[Jotzo 2002] JOTZO, Joachim: *Aktive Landmarken zur Positionsbestimmung von autonomen Fahrzeugen*, "Fakultät für Elektrotechnik und Informationstechnik der Technischen Universit¨at Chemnitz", Dissertation, 2002. – URL http://archiv.tu-chemnitz.de/pub/2002

[Kind 2005] KIND, Andreas: *Positionsbestimmung*, Universität Koblenz-Landau, Seminar, 2005. – URL www.uni-koblenz.de/~zoebel/ws2004/Positionsbestimmung.pdf

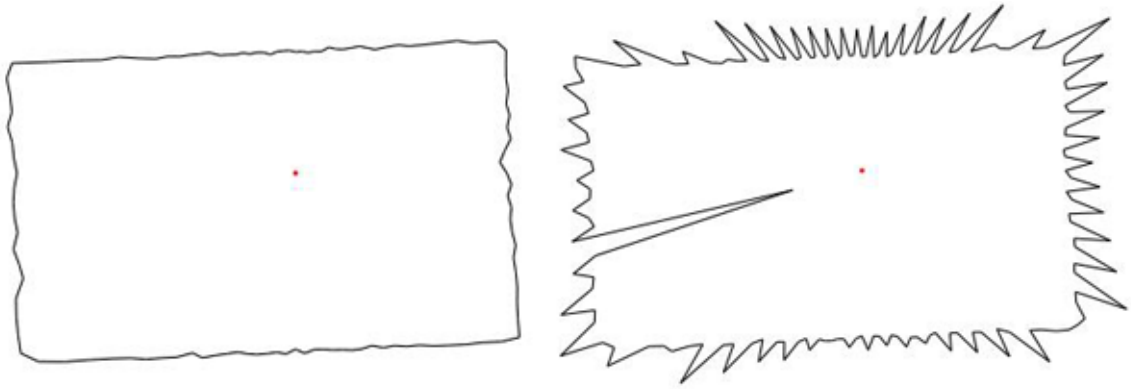
[Papula 1998] PAPULA, Lothar: *Mathematik für Ingenieure und Naturwissenschaftler*. Vieweg, 1998. – ISBN 3-528-74236-4

[Ries 2005] RIES, Sascha: *Hinderniserkennung*, Universität Koblenz-Landau, Seminar, 2005. – URL www.uni-koblenz.de/~zoebel/ws2004/Hinderniserkennung.pdf

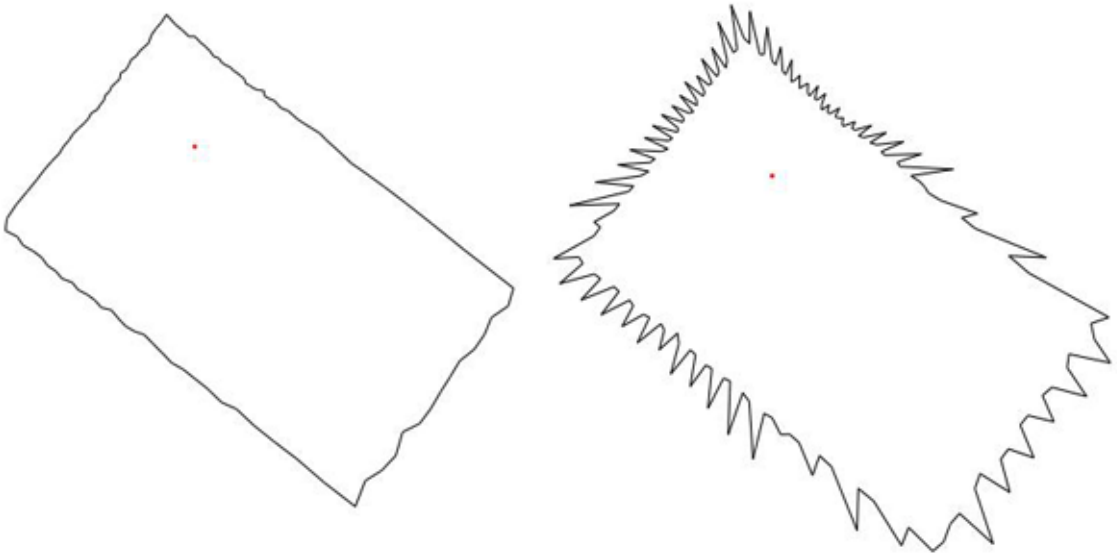
[Schreibern 2007] SCHREIBERN, Carsten: *Telemetrie in mobilen Systemen mit TCP/IP, WLAN und Feldbus Technologien*, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2007. – URL http://users.informatik.haw-hamburg.de

[WEDICO 2001] WEDICO: *mathe-lexikon.at*. 2001. – URL www.wedico.de

A Anhang



Winkel=180°; X=40cm; Y=50cm



Winkel=unknow; X=unknow; Y=unknow

Abbildung A.1: Konturen des Raums Abb-3

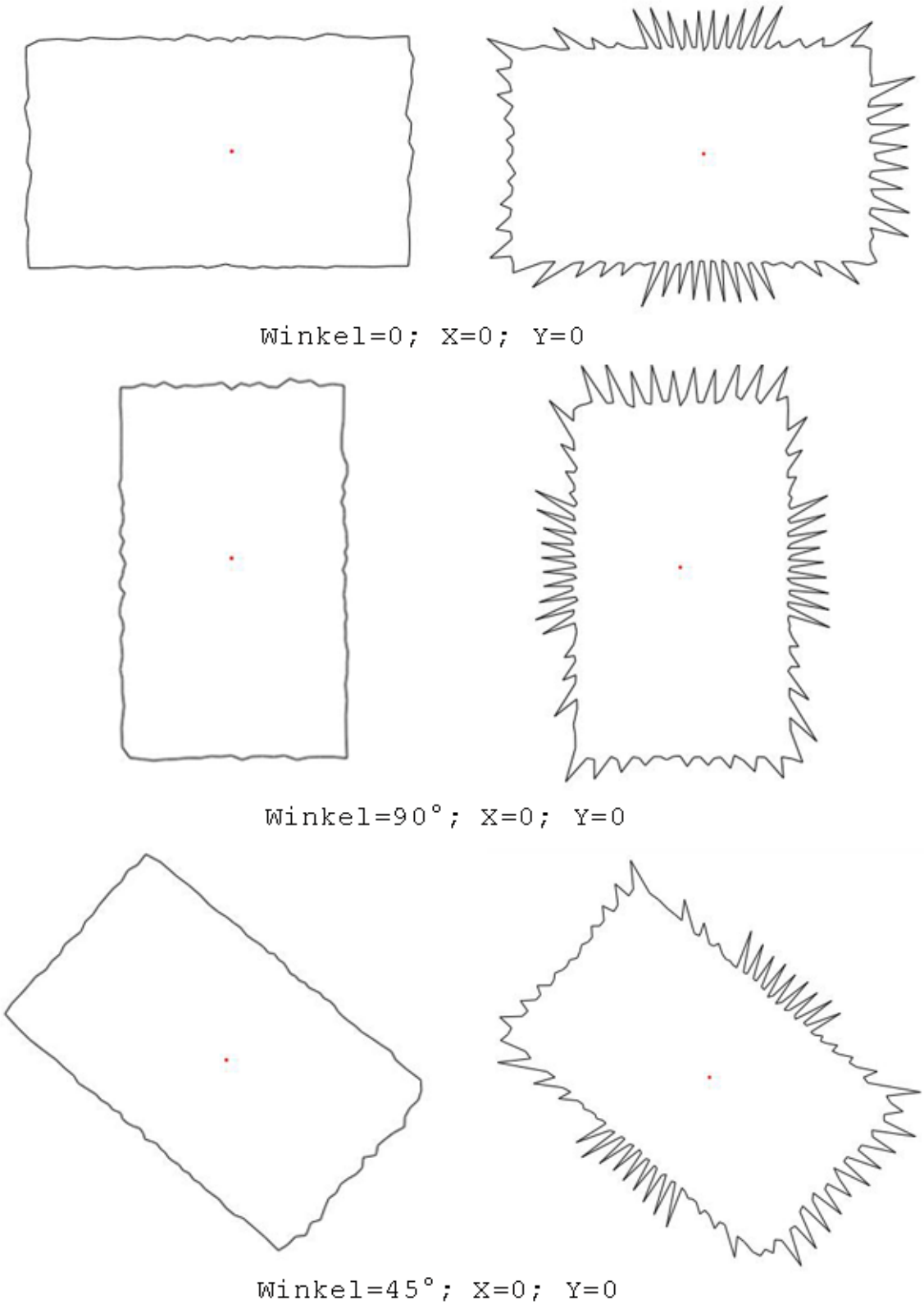
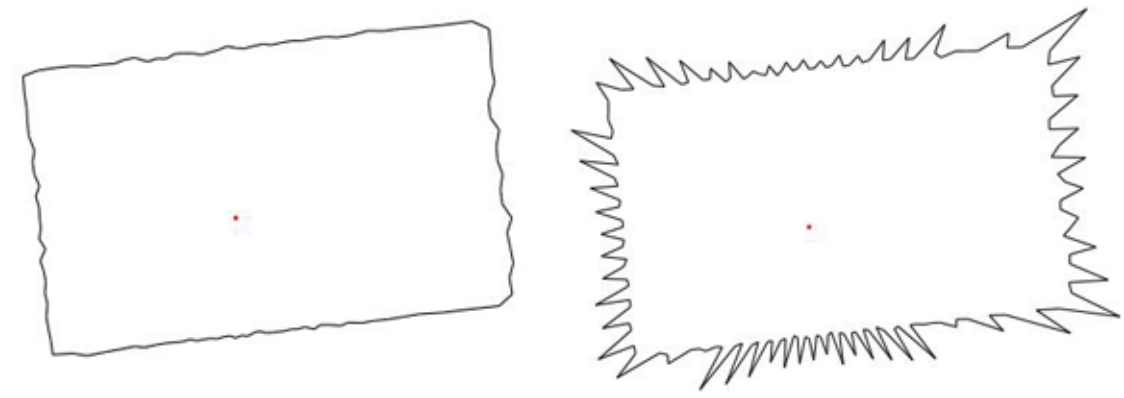
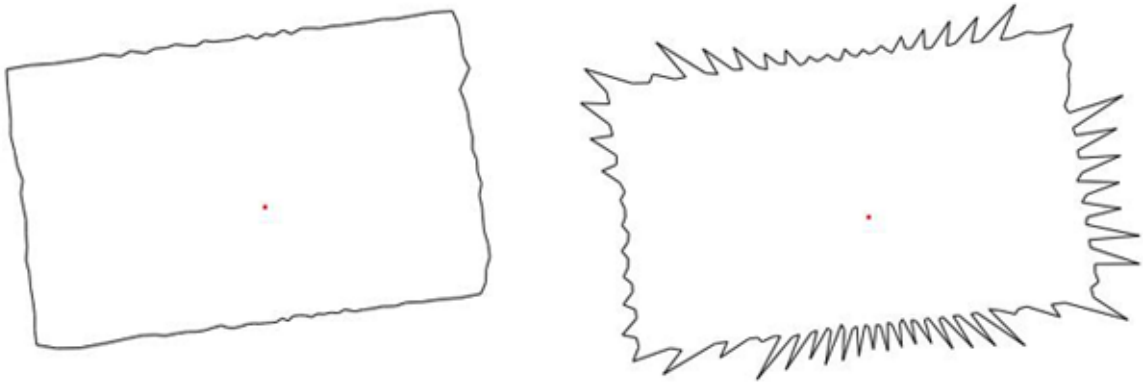


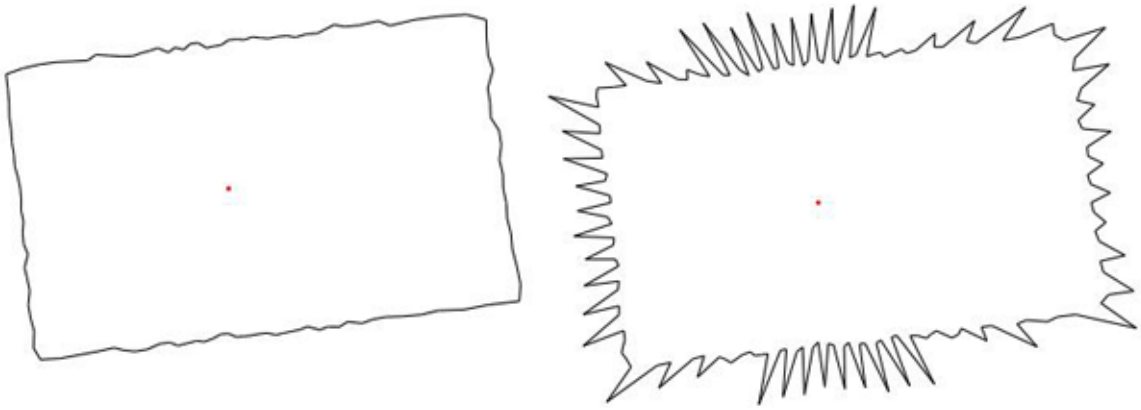
Abbildung A.2: Konturen des Raums Abb-1



Winkel=0; X=40cm; Y=50cm



Winkel=0; X=0; Y=50cm



Winkel=0; X=40cm; Y=0

Abbildung A.3: Konturen des Raums Abb-2

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 23. April 2008	
Ort, Datum	Unterschrift