DIPLOMARBEIT

Steuerung eines fahrbaren Roboters mittels zielverfolgender Totalstation

Ausgeführt am Institut für Geodäsie und Geophysik

unter der Anleitung von

Univ.-Prof. Dipl.-Ing. Dr. techn. Andreas Wieser

Univ.-Ass. Dipl.-Ing. Stefan Lederbauer

Stefan Lackner

Iglbach 15

4171 Auberg

15. Februar 2012 _____

Danksagung

Es ist mir ein Anliegen, all jenen Personen Dank zu sagen, die mich bei der Erstellung meiner Diplomarbeit unterstützt haben. Vor allem danke ich meinen Betreuern Prof. Dr. Andreas Wieser und Dipl.-Ing. Stefan Lederbauer für die Unterstützung, die zahlreichen Tipps und Hilfestellungen während der Durchführung und für die Begutachtung der Arbeit. Spezieller Dank gilt auch Herrn Dipl.-Ing. Lederbauer für die vielen technischen Vorarbeiten und die Konfiguration der mobilen Roboterplattform.

Ich bedanke mich bei der Forschungsgruppe Ingenieurgeodäsie und deren MitarbeiterInnen unter der Leitung von Prof. Dr. Wieser für die Bereitstellung des Arbeitsplatzes und des für die Arbeit notwendigen Messinstrumentariums. Vor allem möchte ich Frau Dipl.-Ing. Dr. Miriam Zámečníková danken, die mir bei der Entwicklung des Regelkreises mit Rat und Tat geholfen hat. Besonders erwähnen möchte ich auch meine Studienkollegin und jetzige Institutsmitarbeiterin, Frau Dipl.-Ing. Eva Maria Buchebner, die immer ein offenes Ohr für meine Anliegen hatte sowie meine Studienkollegen Roland, Dominik und Wolfgang.

Abschließend möchte ich mich ganz herzlich bei meinen Eltern und meiner Familie bedanken für die finanzielle und moralische Unterstützung während der gesamten Studienzeit.

Kurzfassung

Der Einsatz zielverfolgender Totalstationen und echzeitkinematikfähiger GNSS-Empfänger für die Führung und Steuerung von Baumaschinen hat in den letzten Jahren die Automatisierung von Bauprozessen und damit eine wesentliche Steigerung der Arbeitseffizienz auf Großbaustellen bewirkt. Geschlossene Regelkreise ermöglichen eine ständige Nachführung einer Vorgabengröße und entlasten den Maschinenoperateur, sodass dieser bei sogenannten „voll-automatischen" 3D-Steuerungssystemen nur noch die Geschwindigkeit regulieren muss.

Im Rahmen dieser Diplomarbeit wurden Algorithmen zur Steuerung der mobilen Roboterplattform der Forschungsgruppe Ingenieurgeodäsie an der Technischen Universität Wien entwickelt, wobei der Steuerrechner per WLAN mit dem roboterinternen Rechner kommuniziert. Der Zusammenhang zwischen den verwendeten Steuerinputs und der tatsächlichen Positions- und Orientierungsänderungen wird durch ein mathematisches Modell beschrieben und erlaubt die Modellierung der fahrdynamischen Eigenschaften des Roboters. Die Kommunikation zwischen dem zur Steuerung verwendeten Rechner und dem Robotercomputer erfolgt dabei über eine TCP/IP-Socketverbindung.

Durch den in MATLAB implementierten Regelkreis wird der Roboter entlang einer durch Koordinaten vorgegebene Soll-Trajektorie navigiert und der Steuerrechner erhält in Echtzeit die Position des am Roboter befestigten 360°-Prismas von einer Totalstation zur Regelung der Querabweichungen. Die Übertragung der Trackingdaten zwischen Totalstation und Steuerrechner erfolgt über die GeoCOM-Schnittstelle. Bei den durchgeführten Testfahrten werden verschiedene Befestigungspositionen des Prismas auf dem Roboter und deren Auswirkung bei Richtungswechseln der Soll-Trajektorie auf die Regelgüte untersucht und die Unterschiede aufgezeigt.

Die Analyse der Abweichungen zwischen der gemessenen und der vorgegebenen Trajektorie zeigt, dass ein Oval bestehend aus zwei Geraden und zwei Halbkreisbögen mit einem rms der lateralen Abweichungen zur Soll-Trajektorie von besser als 7 mm nachgefahren werden kann.

Abstract

In the last couple of years the application of target tracking total stations and real-time kinemetic GNSS receivers for the guidance and control of construction machines has led to an automation of construction phases and therefore to a significant increase of working efficiency at constructions sites. Closed loop controls allow a permanent setpoint tracing and also relieve the operator during the working process, so that he only has to regulate the velocity of so-called „fully automatic" 3D-control systems.

During my diploma thesis algorithms were developed to control the mobile robot of the research institute of engineering geodesy at the Vienna University of Technology. The control computer communicates through wireless LAN with the internal computer of the robot. The relations between the control inputs and the actual changes in position and orientation are described by using a mathematical model which includes the dynamic properties of the robot. The communication between the computer which is used for controlling and the computer of the robot is established through a TCP/IP-socket connection.

By the implemented closed loop control the robot is being navigated along a trajectory given by coordinates and gets the position of the mounted 360° prism from the total station to control the lateral deviations in real-time. The transmission of tracking data between total station and control computer is established through the GeoCOM interface. During the realised test drives several different positions to fix the prism on the robot and the impact of changes in direction on the control quality are investigated and the differences are shown.

The analysis of deviations between the surveyed and the given trajectory demonstrate that an oval of two straight lines and two semi-circular arches can be followed with a root mean square (rms) of the lateral deviations to the given trajectory of better than 7 mm.

Inhaltsverzeichnis

1 Einleitung

Durch die instrumentelle Weiterentwicklung im Bereich zielverfolgender Tachymeter und echzeitkinematikfähiger GPS-Empfänger etwa Mitte der 90er-Jahre hat sich die Führung und Steuerung von Baumaschinen als wichtiges Teilgebiet in der Ingenieurgeodäsie etabliert (Stempfhuber und Ingensand, 2008).

Zuvor waren Bauverfahren im Tiefbau wie etwa Tunnel-, Gleis-, Flughafen- oder Straßenbau mit statischen Absteckungsverfahren und der mechanischen Abtastung entlang von Führungsleitdrähten durchgeführt worden. Diese Abtastung kann auch von natürlichen Referenzhöhen wie zum Beispiel einer Bordsteinkante oder einer abgefrästen Straßendecke erfolgen, wobei auch akustische Ultraschallsensoren zum Einsatz kommen. Bei optischen Sensorsystemen dient eine von einem Rotationslaser erzeugte horizontale oder leicht geneigte Ebene als direkte Höhenreferenz. Meist werden diese Sensoren zusätzlich mit Neigungssensoren kombiniert. Abbildung 1.1 zeigt einen Straßenfertiger der Firma Joseph Vögele AG bei der mechanischen Abtastung eines Führungsleitdrahtes zur Höhen- und Neigungsregulierung.

Abb. 1.1: Straßenfertiger *Voegele S2001-2* mit mechanischer Abtastung eines Führungsleitdrahtes (www.voegele.info)

Die Genauigkeitsanforderungen liegen je nach Anwendungsgebiet und Baumaschine zwischen 5 und 30 mm in der Höhe und zwischen 5 und 50 mm in der Position, wobei die Arbeitsgeschwindigkeit zwischen 3 m/min und 35 km/h variieren kann. Zwar bieten GNSS-Empfänger bei entsprechender Systemkonfiguration eine sehr einfache Anwendung. Die erreichbare Höhengenauigkeit liegt aber zumeist im Bereich von 1-2 cm, bei Echtzeitanwendungen sogar von mehreren Zentimetern (Stempfhuber und Ingensand, 2008). Präzise Positionsbestimmungen langsam bewegter Objekte im Bereich von 5-10 mm können daher oft nur durch die Verwendung zielverfolgender Robottachymeter erreicht werden. Genauigkeitseinschränkungen insbesondere durch die zeitliche Synchronisation der einzelnen Messsensoren und die Verwendung eines 360°-Prismas werden in Abschnitt 6.3 aufgegriffen.

Als Pilotprojekte für den Einsatz zielverfolgender Tachymeter in der Baumaschinensteuerung ohne Verwendung eines Führungsleitdrahtes gelten die 3D-Steuerung eines Gleitschalenfertigers der ICE-Neubaustrecke zwischen Frankfurt und Köln im Jahr 1999 beziehungsweise die Steuerung der Asphaltfertiger und Fräsen bei einem Autobahnabschnitt zwischen Wil und St. Gallen (siehe Abbildung 1.2).

Abb. 1.2: Pilotprojekte zur 3D-Steuerung von Baumaschinen: Gleitschalungsfertiger der ICE-Neubaustrecke zwischen Frankfurt und Köln, 1999 (links, Stempfhuber und Ingensand, 2008). Asphaltfertiger beim Autobahnabschnitt zwischen Wil und St. Gallen, 2000 (rechts, Zimmermann und Konrad, 2003)

Mit dem Einsatz von Maschinensteuerungssystemen erfolgt eine Optimierung der Produktionsabläufe auf Baustellen, da zum Beispiel vorbereitende Absteckungsarbeiten und die abschließenden Kontrollmessungen wegfallen. Durch die Steigerung der Arbeitseffizienz können auch die Maschinen- und Betriebsstoffkosten gesenkt werden.

1.1 Unterteilung in Führungs- und Steuerungssysteme

Nach Stempfhuber und Ingensand (2008) muss zwischen Führungssystemen (auch Leitsysteme genannt) und Steuerungssystemen unterschieden werden.

Führungssysteme/Leitsysteme (Indicate or Guidance Systems)

Dem Maschinenfahrer werden visuelle Informationen zum manuellen Betrieb der Baumaschine beziehungsweise des Werkzeugs, wie etwa einer Baggerschaufel, auf einem Display im Führerhaus bereitgestellt. Es erfolgt kein Eingriff in den Hydraulikkreislauf der Maschine, die Steuerung erfolgt weiterhin manuell und die Genauigkeit der Ausführung ist daher vom Operateur abhängig. Die Informationen können Höhen oder Neigungen (1D) sein oder 3D-Koordinaten.

Semi-automatische Systeme (Control Systems / Grade Control)

Bei semi-automatischen Systemen muss der Operateur die Maschine selbst fahren und lenken. Dazu wird die 2D-Maschinenposition in Bezug zu den Designdaten grafisch dargestellt. Die Höhensteuerung mit entsprechender Querneigung des Werkzeuges (zum Beispiel die Schar eines Graders) erfolgt automatisch durch einen geschlossenen Regelkreis. Hierfür übernimmt ein Controller die Kommunikation zwischen Messsystem und Hydraulik, die Höhenabweichungen zwischen dem Soll- und Ist-Wert werden als Stellgrößen mit entsprechender Filterung an den Hydraulikkreislauf übergeben.

Voll-automatische 3D-Steuerungssysteme (3D-Control)

Bei 3D-Steuerungssystemen erfolgt neben der Höhen- auch die Richtungssteuerung automatisch, wobei der 3D-Soll-Ist-Vergleich die Hydraulikgrößen definiert. Der Maschinenfahrer regelt nur noch die Geschwindigkeit. Anwendungen dieser Art mit geforderten Genauigkeiten im Bereich von 5 mm stellen einerseits hohe Ansprüche an das Messinstrumentarium, andererseits müssen alle integrierten Sensoren ausreichend genau synchronisiert werden.

Die Grundlage sowohl bei der semi- als auch bei der voll-automatischen Steuerung bildet ein geschlossener Regelkreis, der in Abbildung 1.3 dargestellt ist und dessen Teilschritte in der Steuerungssoftware sequentiell abgearbeitet werden. Dabei wird die von der Sensorik

(Totalstation, Längs- und Querneigungssensoren, Orientierungssensor) gemessene 3D-Position mit dem Design abgeglichen und Korrekturwerte bestimmt. Nach der Übertragung der Abweichungen für die Höhen- und Lagesteuerung an die Maschinenhydraulik erfolgt die Steuerung der Bohle oder des Fräskopfes unter Berücksichtigung des Designs.

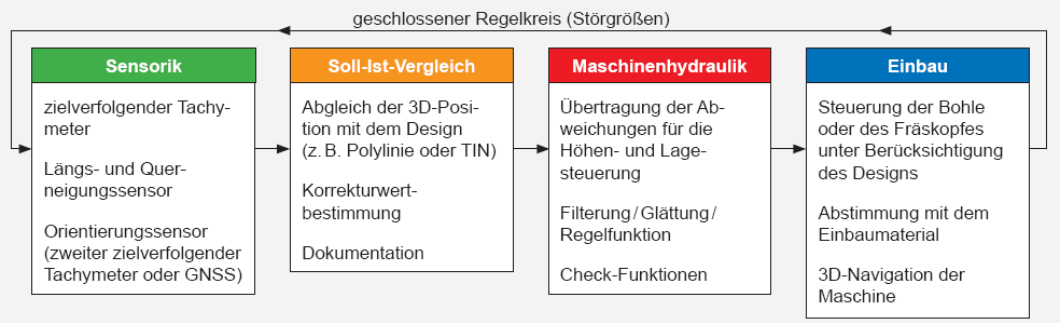


Abb. 1.3: Geschlossener Regelkreis am Beispiel einer voll-automatischen 3D-Steuerung (Stempfhuber und Ingensand, 2008)

1.2 Ziele und Rahmenbedingungen

Das Ziel dieser Arbeit ist die Entwicklung und Implementierung einer Steuerung der mobilen Roboterplattform „Seekur Jr" der Forschungsgruppe Ingenieurgeodäsie an der Technischen Universität Wien als Basis für weiterführende Arbeiten. Der Zugriff auf den Roboter, der in Abbildung 1.4 dargestellt ist, soll dabei von einer MATLAB-Software erfolgen, die über WLAN mit dem roboterinternen Rechner kommuniziert.

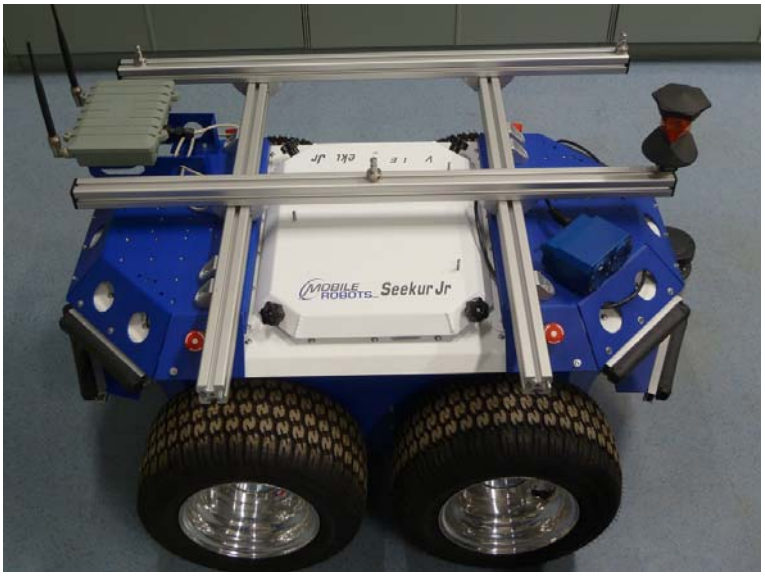


Abb. 1.4: „Seekur Jr" von MobileRobots

Durch die Herleitung eines mathematischen Modells soll die korrekte Umsetzung der Steuerbefehle überprüft und eine Möglichkeit zur Beschreibung der fahrdynamischen Eigenschaften geschaffen werden.

In weiterer Folge soll ein Regelkreis entwickelt werden, um den Roboter mit einer Genauigkeit von 1 cm (1σ, lateral) entlang einer durch Koordinaten vorgegebenen 2D-Soll-Trajektorie zu navigieren. Als Eingangsdaten dienen die mit einer Totalstation gemessenen Messwerte, die in der MATLAB-Software verarbeitet und in Steuerbefehle für den Roboter umgesetzt werden sollen.

Zuletzt soll ein Ausblick über mögliche Weiterentwicklungen des Regelkreises für weiterführende Arbeiten gegeben werden.

1.3 Frühere Arbeiten

Im Rahmen seiner Diplomarbeit hat Dipl.-Ing. Alexander Beetz an der Universität Stuttgart einige Komponenten für einen Regelkreis zur Fahrzeugsteuerung entwickelt. Als Modell-Fahrzeug stand dazu ein Lastwagen im Maßstab von 1:14 zur Verfügung. Im Testsystem (siehe Abbildung 1.5) steuerte der Operateur den Lastwagen mittels der Funkfernsteuerung entlang einer Soll-Trajektorie, ohne dass er diese in der Örtlichkeit einsehen

konnte. Die dementsprechenden Korrekturwerte wurden in einem „Man Machine Interface" grafisch dargestellt. Die Positionsbestimmung erfolgte mit einem Tachymeter (Trimble 5601 DR200+) und einem am Fahrzeug befestigten 360°-Mini-Prisma, wobei sich der Lastwagen in den durchgeführten Testfahrten mit einer Geschwindigkeit von 13-18 cm/s fortbewegte. Durch die ähnlichen vorhandenen Systemkomponenten und dem gleichen Ziel, nämlich der Entwicklung eines Regelkreis zur Steuerung eines Modellfahrzeuges entlang einer 2D-Soll-Trajektorie wurde die Arbeit von Dipl.-Ing. Beetz als Referenzprojekt ausgewählt und wird in diesem Abschnitt näher behandelt.

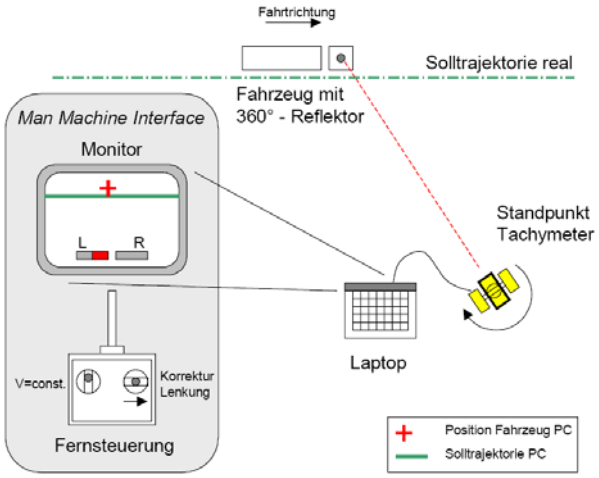


Abb. 1.5: Aufbau Testsystem mit Modell-Lastwagen (Maßstab 1:14) an der Universität Stuttgart (Beetz und Schwieger, 2007)

Die Soll-Trajektorie ist in Abbildung 1.6 dargestellt und beschreibt einen Rundkurs mit der Länge von 11 m bestehend aus zwei Geraden, vier Klothoiden (Klothoidenparameter A = 2) und zwei Kreisbögen mit einem Radius von jeweils 3 m. Für die geometrische Darstellung der Soll-Trajektorie wurden Kleinpunktkoordinaten mit einem Punktabstand von 0,2 m berechnet.

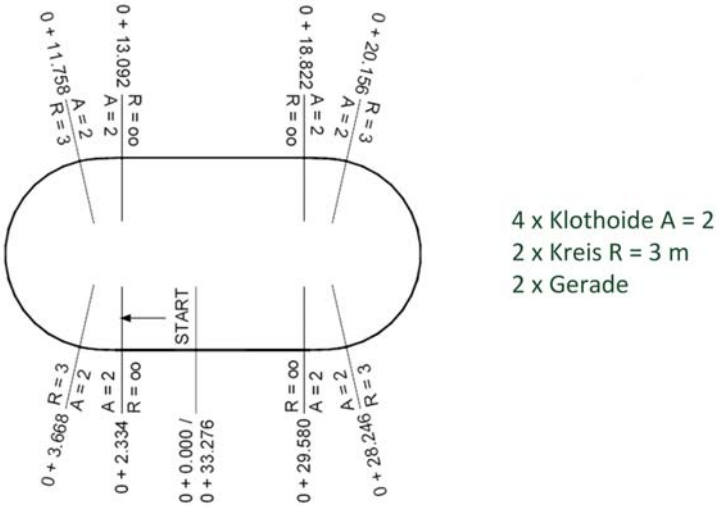


Abb. 1.6: Trassenentwurf (aus Beetz und Schwieger, 2007)

Bei der Steuerung des Lastwagens wurde mithilfe der bekannten Trajektorie der Lenkeinschlag des Fahrzeugs bestimmt, der dem Operateur am „Man Machine Interface"-Monitor angezeigt wurde. Die Berechnung des Lenkeinschlags erfolgte wie in Abbildung 1.7 dargestellt auf Basis des „Kinematic Single Track Model". Dabei kann die Geometrie eines zweispurigen Fahrzeuges vereinfachend als Einspurmodell modelliert werden und die Fahrwege beider Räder werden zu einer zentralen Längsachse in Fahrtrichtung zusammengefasst (Mitschke und Wallentowitz, 2004). Diese rein geometrischen Herleitungen gelten allerdings nur für geringe Geschwindigkeiten und unter der Annahme, dass kein Schlupf der Räder auftritt.

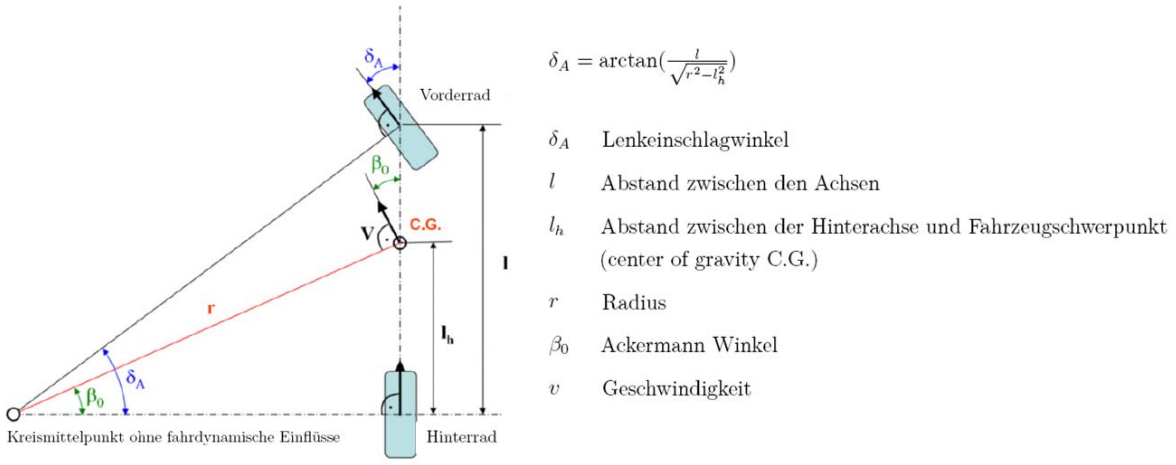


Abb. 1.7: Mathematische Beziehungen des „Kinematic Single Track Model" (nach Mitschke und Wallentowitz, 2004)

Für den Regelkreis wurde zunächst ein Drei-Punkt-Regler verwendet, da nach Merz und Jaschek (2003, S. 114-118) ein stetiger PI-Regler näherungsweise mit einem unstetigen und leichter zu implementierenden Drei-Punkt-Regler umgesetzt werden kann. Die Übertragung der Regelabweichung auf die Stellgröße erfolgt dabei wie in Abbildung 1.8 gezeigt in drei Stufen. Im Nullbereich wird der Wert der Stellgröße nicht verändert. Erst wenn die Regelabweichung den Grenzwert $\pm e_g$ überschreitet, wird die Stellgröße auf $\pm u_g$ eingestellt.

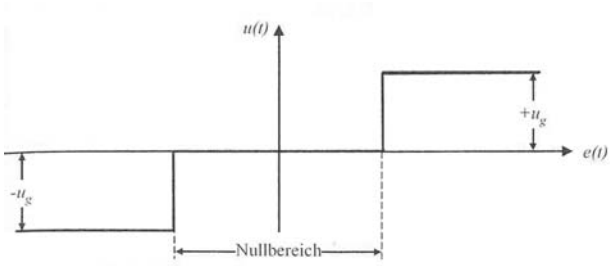


Abb. 1.8: Kennlinie des Drei-Punkt-Reglers (Beetz und Schwieger, 2007)

Die Regelgüte wird als Standardabweichung über die Abweichungen der gemessenen Trajektorie gegenüber der Soll-Trajektorie berechnet. Bei den Testfahrten wurden jeweils zehn Runden des Ovals (siehe Abbildung 1.6) abgefahren und dabei Standardabweichungen zwischen 4,1 und 10,3 mm erreicht. Durch einen im Regelkreis integrierten Kalmanfilter und die Vorausberechnung des Vorderradeinschlagwinkels aus der Soll-Trajektorie konnten diese Werte auf 3,3 bis 4,7 mm verbessert werden. Weitere Details zu den Ergebnissen können bei Beetz und Schwieger (2007) nachgelesen werden.

2 Seekur Jr als Testfahrzeug

Der Messroboter „Seekur Jr" der Firma MobileRobots wurde 2009 von der Forschungsgruppe Ingenieurgeodäsie an der Technischen Universität Wien angekauft. Durch seinen panzerähnlichen, differentiellen Antrieb und seine robuste Bauweise ist er vor allem für Outdoor-Anwendungen sehr gut geeignet und kann Steigungen von bis zu 75 Prozent bewältigen. Drei Nickel-Metall Hybrid (NiMH) Akkus ermöglichen eine Laufzeit von knapp drei Stunden bei ständiger Belastung. Der Roboter besitzt ein Eigengewicht von etwa 70 Kilogramm bei 50 Kilogramm zusätzlicher Tragkraft, erreicht eine maximale Geschwindigkeit von 1,2 Metern pro Sekunde und eine Rotationsgeschwindigkeit um seine eigene Achse von 100 Grad pro Sekunde. Die geometrischen Abmessungen sind in Abbildung 2.1 und Abbildung 2.2 dargestellt.

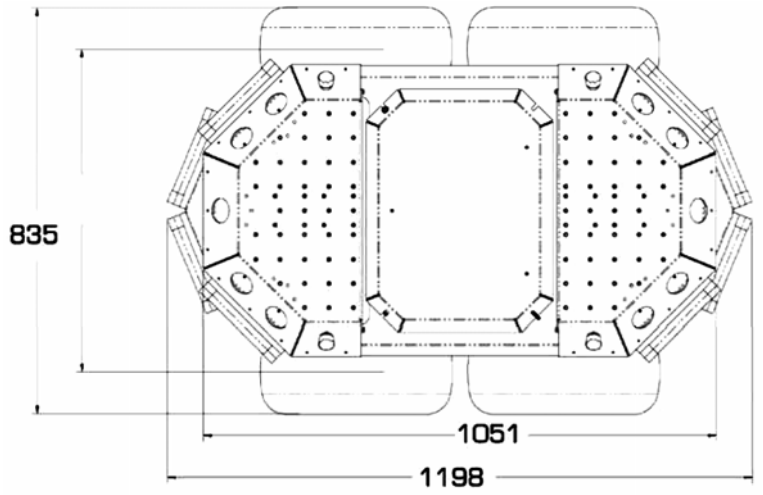


Abb. 2.1: Abmessungen Seekur Jr, Draufsicht (alle Angaben in Millimeter, aus Seekur Jr Operations Manual, MobileRobots, 2010)

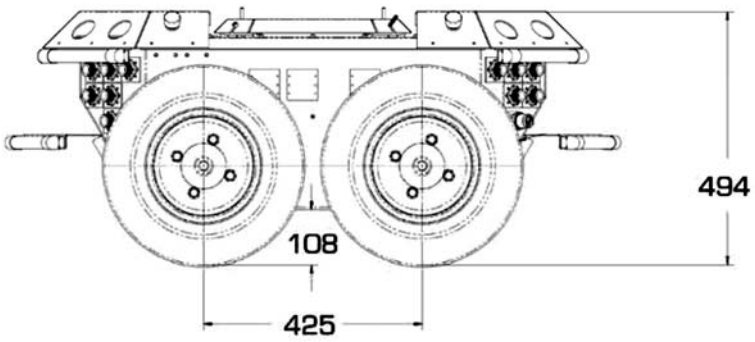


Abb. 2.2: Abmessungen Seekur Jr, Seitenansicht (alle Angaben in Millimeter, aus Seekur Jr Operations Manual, MobileRobots, 2010)

Gesteuert werden kann der Roboter entweder manuell mit einem Joystick oder über den internen Board-Rechner. Dieser interne Computer verfügt über zwei 10/100 Base-T Internet- und vier serielle Anschlüsse, zwei USB-Ports sowie acht analoge Ports. Durch diese Vielzahl an Anschlussmöglichkeiten kann der Roboterrechner an einen Monitor angeschlossen werden und mit Maus und Tastatur bedient werden. Weiters kann von einem externen Computer oder Laptop eine Remote-Verbindung über ein Netzwerkkabel beziehungsweise den integrierten WLAN-Router hergestellt werden.

Roboterbewegungen (Positions- und Orientierungsänderungen) werden mit Odometern aufgezeichnet. Darüber hinaus sind ein Laserscanner (Sick LMS 111-10100) sowie eine Stereokamera montiert und mit dem internen Rechner verbunden.

2.1 Steuerung mit ARIA (Advanced Robotics Interface for Applications)

MobileRobots stellt mit ARIA (Advanced Robotics Interface for Applications) äußerst komplexe Klassenbibliotheken zur Steuerung des Roboters und Bedienung der Messsensoren zur Verfügung. ARIA wurde in der Programmiersprache C++ entwickelt, ist dadurch objektorientiert und ermöglicht das Erstellen von Single- oder Multithreadinganwendungen auf Clientseite, deren Anwendung noch genauer in Abschnitt 6.4 behandelt wird. Durch die Verbreitung von ARIA unter dem GNU General Public License Programm [1], einem freien Software-Lizenz-Projekt, ist der gesamte Quellcode öffentlich zugänglich. Darüber

[1] http://www.gnu.org

hinaus ist es Anwendern auch erlaubt, Änderungen am Quellcode vorzunehmen, wenn sie diesen unter Einhaltung der gleichen Lizenzbestimmungen publizieren. Ein großer Vorteil dieser Programmierschnittstelle liegt in der möglichen Nutzung der Roboterbefehle auf unterschiedlichen Stufen, also von Einzelbefehlen bis hin zu vordefinierten komplexen Bewegungsabläufen.

Zum Zweck von Testfahrten und der Steuerung des Roboters entlang einer Soll-Trajektorie über die MATLAB-Steuersoftware (siehe Kapitel 5) wird auf die sogenannten „Simple Motion Commands" zurückgegriffen. Solche einfachen Bewegungsbefehle wie zum Beispiel *setDeltaHeading* (Drehung um die eigene Achse um einen bestimmten Winkel) oder *setRotVel* (Setzen der Rotationsgeschwindigkeit) erlauben eine sehr direkte Antriebssteuerung und somit auch verschiedene Möglichkeiten zur Bewegungsmodellierung. Eine lineare Bewegung kann so etwa durch die Verwendung des Befehls *move* (und Angabe der zu fahrenden Distanz in Millimetern) oder mittels Setzen der Translationsgeschwindigkeit (*setVel*) in Millimetern pro Sekunde, welche dann eine gewisse Zeit ausgeführt wird (*sleep*), realisiert werden. Für eine Rotationsbewegung am Stand stehen analog dazu die Befehle *setDeltaHeading* beziehungsweise *setRotVel* und *sleep* zur Verfügung.

Eine sehr einfache Möglichkeit für Kurvenbewegungen ist in ARIA mit der Funktion *setVel2* realisiert. Dabei werden für die beiden Radpaare rechts und links jeweils unterschiedliche Antriebsgeschwindigkeiten eingestellt, wodurch sich die Kurvenfahrt ergibt. Leider wird dieser Befehl beim Modell „Seekur Jr" an der TU Wien vom Hersteller MobileRobots nicht unterstützt und ist nur bei der Nachfolgerversion „Seekur" verfügbar. Die Kurvenfahrt des Roboters muss daher aus einer linearen und einer Drehbewegung zusammengesetzt werden. Das wird über eine geeignete Kombination der Befehle *setVel*, *setRotVel* und *sleep* ermöglicht.

Die Verwendung des *sleep*-Befehls in C++ hat allerdings den Nachteil, dass die tatsächliche Bewegungsausführungsdauer des Roboters nicht mit der vom *sleep*-Befehl definierten Zeit übereinstimmt und zusätzlich zeitliche Variationen aufweist. Somit wirkt sich dieser Effekt nachteilig bei der Aneinanderreihung von mehreren gleichen Befehlen aus. Die daraus resultierenden Probleme und geänderten Anforderungen an die Robotersteuerung werden in Abschnitt 6.5 erneut aufgegriffen.

2.2 Rahmenkonstruktion für Prismenhalterung

Standardmäßig verfügt Seekur Jr keine Möglichkeit zur Befestigung von Prismen beziehungsweise Prismenhalterungen. Daher wurde am Institut für Geodäsie und Geophysik der TU Wien ein modulares Profilsystem vom deutschen Hersteller Item angeschafft und als Rahmenkonstruktion am Roboter montiert. Der große Vorteil der Konstruktion liegt in der Verschiebbarkeit der einzelnen Schienen und der vielseitigen Verwendungsmöglichkeit für mehrere Prismen, GPS-Antennen, usw. Abbildung 2.3 zeigt den Roboter mit der montierten Rahmenkonstruktion und einem 360°-Prisma vorne über der Mittelachse.

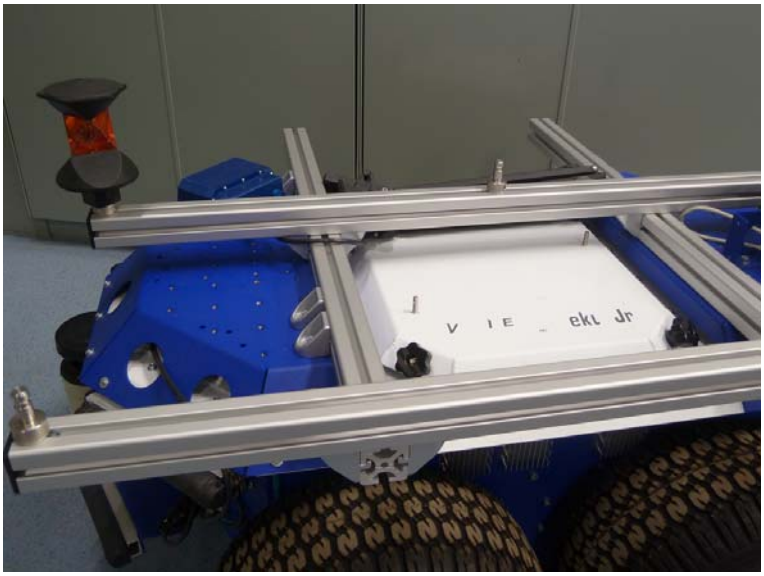


Abb. 2.3: Seekur Jr mit der montierten Item-Rahmenkonstruktion für die Prismenhalterung

3 Steuerungs- und Regelungstechnik

Die Steuerungs- und Regelungstechnik als zentraler Bestandteil der Automatisierungstechnik beschäftigt sich mit der automatisierten Steuerung und Regelung zahlreicher technischer Produkte, zum Beispiel in den Bereichen Haushalt, Medizin- oder Kraftfahrzeugtechnik. Im Unterschied zur Steuerungstechnik findet bei der Regelungstechnik eine Rückkopplung statt, bei der ein Teil der Ausgangsgröße auf den Eingang des Systems zurückgeführt wird. Dieser Abschnitt stützt sich weitgehend auf Tröster (2011) und Lunze (2008).

Regelkreise werden in der Regelungstechnik bei Anlagen, Maschinen und Prozessen eingesetzt, um den Mensch Kontrollaufgaben abzunehmen. „Das Regeln - die Regelung - ist ein Vorgang, bei dem eine Größe, die zu regelnde Größe (Regelgröße), fortlaufend erfasst, mit einer anderen Größe, der Führungsgröße, verglichen und im Sinne einer Angleichung an die Führungsgröße beeinflusst wird" (Tröster, 2011 nach DIN 19226). Bei der Regelung wird also wie in Abbildung 3.1 dargestellt ein Soll-Ist-Wertevergleich durchgeführt, wodurch sich die Regelgröße über den geschlossenen Wirkungsweg des Regelkreises selbst fortlaufend beeinflusst. Das Ziel der Regelung ist es, die gemessene oder aus Messwerten abgeleitete Regelgröße $y(t)$ der vorgegebenen Führungsgröße $w(t)$ so nachzuführen, dass die Regeldifferenz $e(t) = w(t) - y(t)$ im Idealfall für alle Zeitpunkte t verschwindet. Die Regeleinrichtung (auch Regler genannt) gibt die Stellgröße $u(t)$ so vor, dass die Regelgröße $y(t)$ auf einem vorgegebenen konstanten Wert geschaltet oder so geändert wird, wie es die Führungsgröße $w(t)$ verlangt. Wird auf die Bestimmung und Rückführung der Regeldifferenz verzichtet, so spricht man von Steuerung.

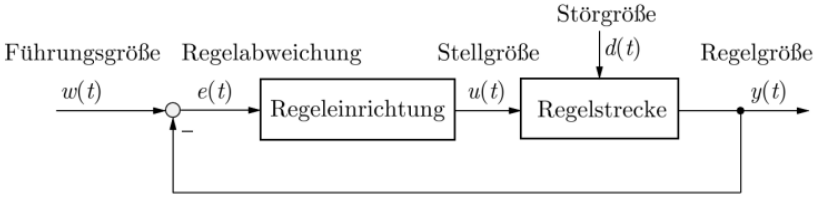


Abb. 3.1: Der standardisierte einschleifige Regelkreis (Lunze, 2008)

Da sowohl die Mess- als auch die Stelleinrichtung dynamische Eigenschaften besitzen, wird der standardisierte einschleifige Regelkreis erweitert und dessen Grundstruktur in Abbildung 3.2 dargestellt. So ist die Regelgröße $y(t)$ meist nur als Messwert $y_m(t)$ verfügbar und mit einem Messrauschen $r(t)$ behaftet. Das Stellglied setzt den vom Regler vorgegebenen Wert $u(t)$ für die Stellgröße in den am Prozess wirksamen Wert $u_R(t)$ um. Im Allgemeinen weisen diese Stelleinrichtungen eigene dynamische Eigenschaften auf und zeigen nichtlineares Verhalten, wodurch sich zum Beispiel untere und obere Schranken für $u_R(t)$ ergeben können. Weiters kann auch eine Ansprechempfindlichkeit auftreten, sodass $u_R(t)$ nur dann verändert wird, wenn sich $u(t)$ um einen Mindestbetrag ändert. Das Reglergesetz beschreibt den Zusammenhang, welche Stellgröße $u(t)$ bei einer gewissen Regeldifferenz $e(t)$ gewählt wird, und hat die allgemeine Form

$$u(t) = k(y(t), w(t)) \tag{3.1}$$

beziehungsweise

$$u(t) = k_R(e(t)). \tag{3.2}$$

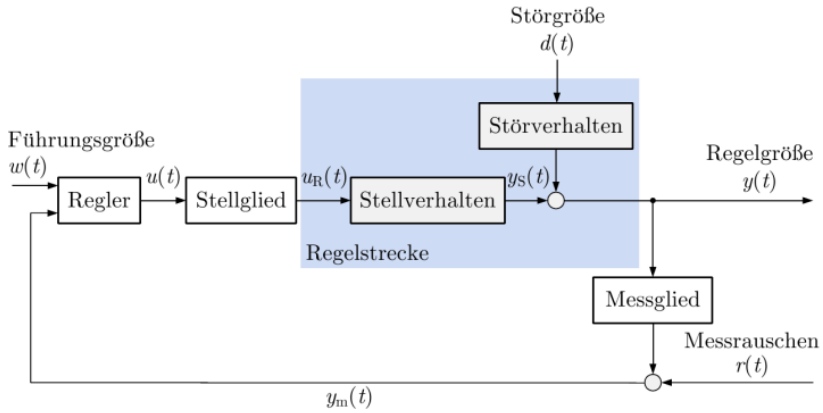


Abb. 3.2: Erweiterte Grundstruktur des Regelkreises (Lunze, 2008)

3.1 Arten von Reglern

Drei Reglerarten sind in der Regelungstechnik von grundlegender Bedeutung: der proportionale (P), der integrale (I) und der differenzierende (D) Regler.

3.1.1 Proportionaler Regler

Beim proportionalen Regler wird folgende Strategie verfolgt: „Je größer die Missachtung bzw. die Abweichung zu einem Gebot ist, desto größer fällt die Strafe bzw. Gegenreaktion aus" (Tröster, 2011). Die Stellgröße $y(t)$ wird also proportional aus der Regeldifferenz $e(t)$ abgeleitet:

$$y(t) = K_p \cdot e(t) \tag{3.3}$$

Durch den Proportionalitätsfaktor K_p reagiert der Regler unmittelbar und schnell auf eine Änderung der Regelgröße. Er kann aber Störungen der Regelstrecke nicht ausregeln, ist deshalb ungenau und hinterlässt eine bleibende Regeldifferenz.

Aufgrund seiner direkten und schnellen Reaktion auf Abweichungen bietet sich der proportionale Regler auch als passende Regeleinrichtung zur lateralen Steuerung des Roboters über eine Soll-Trajektorie an.

3.1.2 Integraler Regler

Bei Verwendung integraler Regler wird versucht, die Summe der Abweichungen zu Null werden zu lassen. Daher gilt: „Je länger eine Abweichung von einem Sollzustand andauert, umso größer fällt die Gegenreaktion aus" (Tröster, 2011). Hier wird also die Regeldifferenz über die Zeit aufsummiert, wodurch sich folgender Zusammenhang für die Stellgröße $y(t)$ ergibt:

$$y(t) = K_i \int_0^t e(t')dt' \tag{3.4}$$

oder:

$$\dot{y}(t) = K_i \cdot e(t) \tag{3.5}$$

Im Gegensatz zum P-Regler stellt der I-Regler die Regelgröße exakt auf die Führungsgröße ein, benötigt für diesen Vorgang aber länger und neigt zum Schwingen.

Bei der Verwendung eines integralen Reglers zur Robotersteuerung müsste der Regler wahrscheinlich so modifiziert werden, dass nur eine Anzahl an Abweichungen in die Berechnung der Stellgröße eingeht, um den Einfluss großer Abweichungen (zum Beispiel zu Beginn der Roboterfahrt) auf das weitere Regelverhalten zu minimieren.

3.1.3 Differenzierender Regler

Der differenzierende Regler verhält sich nach folgendem Prinzip: „Je schneller eine Abweichung entsteht, umso größer fällt die Gegenreaktion aus" (Tröster, 2011). Er reagiert nur auf zeitliche Änderungen der Regeldifferenz $e(t)$ durch ein der Änderungsgeschwindigkeit proportionales Stellsignal:

$$y(t) = K_d \cdot \dot{e}(t) \tag{3.6}$$

Aus diesem Regelverhalten resultiert, dass der D-Regler zeitlich konstante Regeldifferenzen nicht ausregelt, egal wie groß diese sind. Deshalb ist diese Art von Regler allein prinzipiell ungeeignet und erlangt seine Wirkung erst durch Kombination mit P- und I-Regelverhalten. Außerdem kann ein Regelkreis durch die Verwendung eines D-Reglers sehr leicht instabil werden, vor allem wenn verrauschte Messsignale vorliegen.

3.1.4 Kombinierte Regler

Die angeführten P-, I- und D- Regler weisen sowohl Vorteile als auch Nachteile bei ihrem Regelverhalten auf. Durch Kombination der einzelnen Basisregler können deren Nachteile reduziert werden. Als Beispiel sei der PID-Regler als Universalregler durch Parallelschaltung von P-, I- und D-Regler angeführt, bei dem sich die Vorteile aller drei Grundregler summieren. „Das Führungsverhalten eines PID-Reglers ist gekennzeichnet durch schnelles Vorhalten (D), gezieltes Anfahren in die Nähe des Sollwerts (P) und abschließendes präzises

Ausregeln der Regeldifferenz (I)" (Tröster, 2011). Diese drei Wirkungseigenschaften können anhand von Abbildung 3.3 verdeutlicht werden, wo die Führungsgröße einen plötzlichen Sprung erfährt. Der D-Regler steuert die Regelgröße beim Führungssprung sehr schnell in Richtung der neuen Führungsgröße, auch wenn er darüber „hinausschießt" (1). Die durch den P-Regler zurückgebliebene Regeldifferenz bei der Annäherung zur Sollgröße (2) beseitigt schließlich der I-Regler (3).

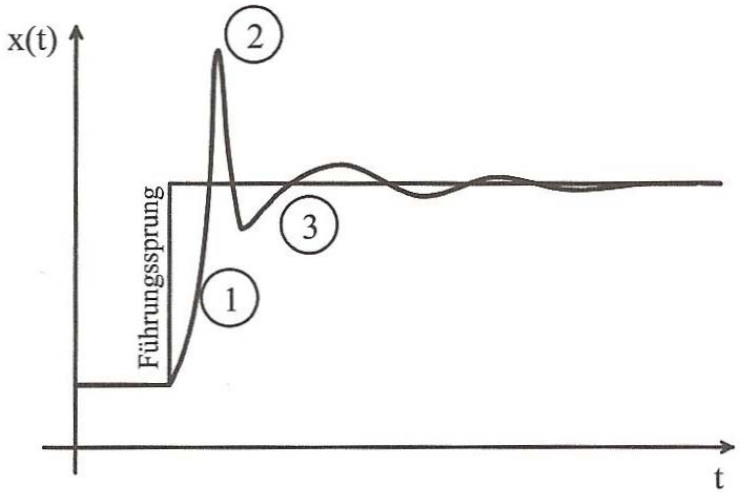


Abb. 3.3: Führungsverhalten des PID-Reglers (Tröster, 2011)

Weitere mögliche Kombinationen und Regelungskonzepte sind für diese Arbeit nicht von Bedeutung und können aus Tröster (2011) oder Lunze (2008) entnommen werden.

3.2 Beurteilung eines Regelsystems

Eine quantitative Beurteilung des Zeitverhaltens von Regelungen wird durch die Regelgüte erreicht, die ein Maß der Abweichung zwischen Regelgröße $x(t)$ und Führungsgröße $w(t)$ darstellt (Tröster, 2011). Dazu wird das Einschwingverhalten eines Regelkreises auf einen endlichen Sprung der Führungsgröße (oder der Störgröße) wie in Abbildung 3.4 betrachtet. Typische Parameter sind dabei die An- und Ausregelzeit T_{An} , T_{Aus} , die Regelfläche I_R , die Überschwingweite $x_{\ddot{U}}$ und die bleibende Regeldifferenz. Die minimale Zeitspanne, welche die Regelgröße $x(t)$ für den Übergang zum Toleranzbereich des neuen Zustands benötigt, wird Anregelzeit T_{An} genannt. Unter der Ausregelzeit T_{Aus} versteht man die Zeitdauer,

die die Regelgröße $x(t)$ für den Übergang vom „alten" in den „neuen" Beharrungszustand braucht und sich in diesem Toleranzbereich einpendelt. Die Regelfläche I_R ist die Summe aller Teilflächen zwischen der Regelgröße $x(t)$ und dem Beharrungswert eines Übergangs bei einem Stör- oder Führungssprung. Als Überschwingweiten $X_{\ddot{U}}$ werden die Ausschläge der Regelgröße $x(t)$ über den „neuen" Beharrungszustand bezeichnet.

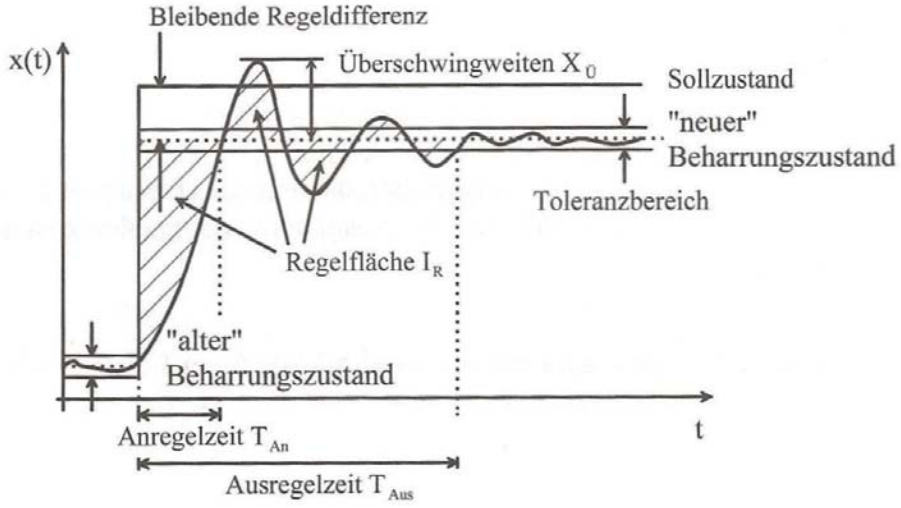


Abb. 3.4: Parameter zur Beurteilung der Regelgüte (Tröster, 2011)

4 Mathematische 2D-Modellierung

Ziel bei der Herleitung eines mathematischen Modells ist die Berechnung der Positions- und der Richtungsänderung des Roboters nach Ausführung einer vorgegebenen Befehlssequenz. Da im Rahmen dieser Arbeit die Querabweichung zu einer vorgegebenen 2D-Trajektorie untersucht werden soll, wird die Position bzw. deren Änderung durch ebene, zweidimensionale Koordinaten und die Orientierung/Richtungsänderung durch einen Horizontalwinkel beschrieben. Die 2D-Positionskoordinaten des Roboters entsprechen dabei im weiteren Verlauf der Ausarbeitung jeweils dem in die Ebene projizierten Zentrum des 360°-Prismas. Durch diese Festlegung kann bei der Herleitung des mathematischen Modells auf ein zusätzliches Roboter-Koordinatensystem verzichtet werden, um den geometrischen Zusammenhang zwischen der Lage des Prismenzentrums und einem fix definierten Punkt am Roboter (wie zum Beispiel dem geometrischen Schwerpunkt) herstellen zu können.

Der Roboter verfügt über zwei Freiheitsgrade: Die Translationsgeschwindigkeit v_T und die Rotationsgeschwindigkeit um die eigene Drehachse ω_D. Die beiden Räder einer Seite sind mechanisch verbunden und führen bis auf Reibungseffekte der Riemen die gleiche Rotationsbewegung aus. Bei der Translation haben das linke und das rechte Riemenpaar den gleichen, bei der Drehung um die eigene Achse einen entgegengesetzten Drehsinn. Unterschiedliche Rotationsgeschwindigkeiten können am Seekur Jr nicht eingestellt werden, weshalb eine Kurvenbewegung durch eine zusammengesetzte Bewegung aus Translation und Rotation erfolgen muss (siehe Abschnitt 2.1).

Weitere dynamische Einflussfaktoren auf die Roboterbewegung wie Schlupf und Drift der Räder werden bei der Herleitung des Bewegungsmodells nicht berücksichtigt.

4.1 Rotation

Für die Rotation am Stand (Drehung um die Mittelachse) steht als Einstellgröße die Winkelgeschwindigkeit am Roboter zur Verfügung. Sie wird in der Einheit °/sec angegeben und in den nachfolgenden Herleitungen mit ω_D bezeichnet; der bei der Rotation überstrichene Drehwinkel mit φ_D. Abbildung 4.1 zeigt den Roboter in vereinfachter Darstellung bei der Rotation um die Mittelachse. Der Abstand r_D vom Drehpunkt zur Mitte jedes der vier Räder ist aus dem Benutzerhandbuch des Herstellers nicht ersichtlich (siehe Abbildung 2.1) und wurde daher direkt am Roboter gemessen (0,384 m).

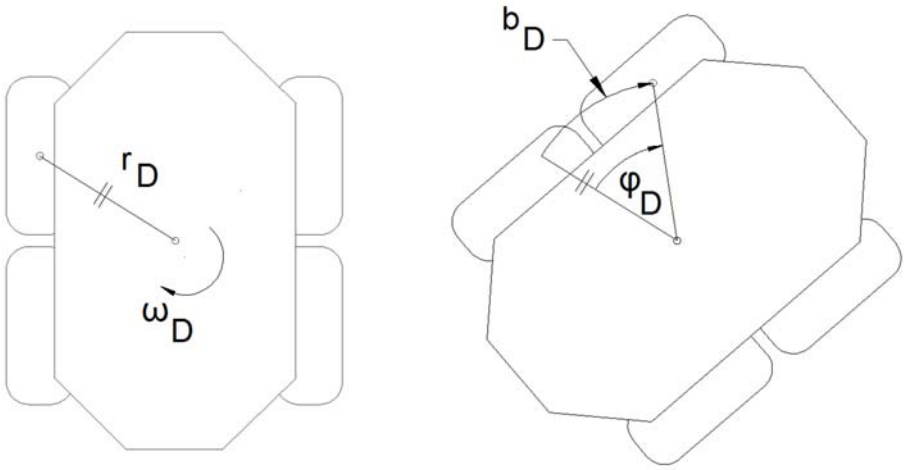


Abb. 4.1: Rotation am Stand (Rechtsdrehung)

$$b_D = \varphi_D \cdot r_D \tag{4.1}$$

$$\varphi_D = \omega_D \cdot t \tag{4.2}$$

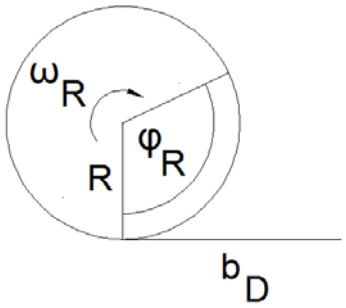


Abb. 4.2: Rotation eines Rades

Betrachtet man die Bewegung eines einzelnen Rades in Abbildung 4.2, kann aus dem zurückgelegten Weg b_D und der Zeit t auf die Winkelgeschwindigkeit ω_R geschlossen werden. Der Radius R der vier Räder beträgt 0,2 m.

$$\varphi_R = \frac{b_D}{R} = \frac{\varphi_D \cdot r_D}{R} \tag{4.3}$$

$$\omega_R = \frac{\varphi_R}{t} = \frac{\varphi_D \cdot r_D}{R \cdot t} = \frac{\omega_D \cdot r_D}{R} \tag{4.4}$$

4.2 Translation und Rotation

Wie bereits in Kapitel 2 erläutert, wird die Kurvenfahrt mit dem Roboter mittels Translation und gleichzeitiger Rotation realisiert. Translationsgrößen werden in den nachfolgenden Formeln mit einem tiefgestellten Index T gekennzeichnet, Rotationsgrößen, wie schon im vorigen Abschnitt, mit dem Index R.

Der Zusammenhang zwischen der Winkelgeschwindigkeit ω_T eines Rades und der daraus resultierenden Translationsbewegung mit der Geschwindigkeit v_T lautet:

$$\omega_T = \frac{v_T}{R} \tag{4.5}$$

v_T steht dabei für die Translationsgeschwindigkeit und kann am Roboter in der Einheit *Millimeter pro Sekunde* eingestellt werden. In der weiteren Herleitung wird eine Rechtsdrehung angenommen, für eine Linksdrehung müssen nur die Indizes $_r$ („rechts") und $_l$ („links") vertauscht werden.

Die Winkelgeschwindigkeit des rechten Rades ω_r resultiert aus der Differenz der Translations-Winkelgeschwindigkeit ω_T und der Rotations-Winkelgeschwindigkeit ω_R, die des linken Rades ω_l aus deren Summe:

$$\omega_r = \omega_T - \omega_R = \frac{v_T}{R} - \frac{\varphi_D \cdot r_D}{R \cdot t} = \frac{v_T \cdot t - \varphi_D \cdot r_D}{R \cdot t} \tag{4.6}$$

$$\omega_l = \omega_T + \omega_R = \frac{v_T}{R} + \frac{\varphi_D \cdot r_D}{R \cdot t} = \frac{v_T \cdot t + \varphi_D \cdot r_D}{R \cdot t} \tag{4.7}$$

Mit Gleichung 4.5 kann auf die jeweiligen Bahngeschwindigkeiten v_r und v_l und in weiterer Folge die zurückgelegten Bogenlängen b_r und b_l der beiden Radpaare geschlossen werden:

$$v_r = \omega_r \cdot R = \frac{v_T \cdot t - \varphi_D \cdot r_D}{R \cdot t} \cdot R \tag{4.8}$$

$$v_l = \omega_l \cdot R = \frac{v_T \cdot t + \varphi_D \cdot r_D}{R \cdot t} \cdot R \tag{4.9}$$

$$b_r = v_r \cdot t = \frac{v_T \cdot t - \varphi_D \cdot r_D}{t} \cdot t = v_T \cdot t - \varphi_D \cdot r_D \tag{4.10}$$

$$b_l = v_l \cdot t = \frac{v_T \cdot t + \varphi_D \cdot r_D}{t} \cdot t = v_T \cdot t + \varphi_D \cdot r_D \tag{4.11}$$

Abbildung 4.3 stellt die Kurvenfahrt der beiden Vorderräder des Roboters mit dem Radius R_{Kv} dar. Das rechte Rad legt dabei den Weg b_r zurück, das linke Rad den Weg b_l. In dieser Abbildung und den folgenden Ableitungen wird vorerst nicht berücksichtigt, dass die Drehung um den Mittelpunkt des Roboters und nicht um einen Punkt auf der Verlängerung der Vorderachse erfolgt. Diese Korrektur erfolgt erst nach der Berechnung von R_{Kv} und ist in Abbildung 4.4 dargestellt.

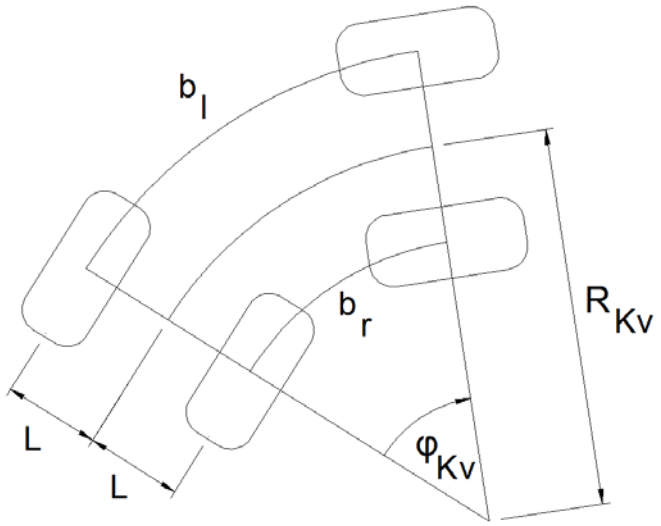


Abb. 4.3: Kurvenfahrt der Vorderräder mit Translationsgeschwindigkeit v_T und gleichzeitiger Rotationsgeschwindigkeit ω_D

Setzt man das Verhältnis zwischen Bogenlängen und deren Kurvenradien gleich, erhält man unter Verwendung von L als halber Spurweite und Umformung R_{Kv} als Radius für die Kurvenfahrt der Vorderräder:

$$\frac{b_l}{R_{Kv} + L} = \frac{b_r}{R_{Kv} - L} \tag{4.12}$$

$$(R_{Kv} - L) \cdot b_l = (R_{Kv} + L) \cdot b_r \tag{4.13}$$

$$R_{Kv} \cdot b_l - L \cdot b_l = R_{Kv} \cdot b_r + L \cdot b_r \tag{4.14}$$

$$R_{Kv} \cdot (b_l - b_r) = L \cdot (b_l + b_r) \tag{4.15}$$

$$R_{Kv} = L \cdot \frac{b_l + b_r}{b_l - b_r} \tag{4.16}$$

Ersetzt man b_r und b_l nach Gleichung 4.10 und Gleichung 4.11 erhält man weiters:

$$R_{Kv} = L \cdot \frac{v_T \cdot t + \varphi_D \cdot r_D + v_T \cdot t - \varphi_D \cdot r_D}{v_T \cdot t + \varphi_D \cdot r_D - v_T \cdot t + \varphi_D \cdot r_D} \tag{4.17}$$

$$R_{Kv} = L \cdot \frac{v_T \cdot t}{\varphi_D \cdot r_D} = L \cdot \frac{v_T \cdot t}{\omega_D \cdot t \cdot r_D} = L \cdot \frac{v_T}{\omega_D \cdot r_D} \tag{4.18}$$

Wie in Abbildung 4.4 angedeutet bewegt sich der Prismenpunkt P_v über der Mitte der Vorderachse bei der Kurvenfahrt mit einem größeren Kurvenradius als der Dreh- und Mittelpunkt des Roboters, da sich ja beide Punkte auf der (fixen) Roboter-Mittelachse befinden. Um den Kurvenradius von Prismenpunkt P und somit den mittleren Kurvenradius des gesamten Roboters zu erhalten, muss man daher noch die Größe R_{Kv} mit dem Abstand d korrigieren. d ist dabei der horizontale Abstand zwischen dem Mittelpunkt P des Roboters und der Prismenposition P_v.

$$R_K = \sqrt{{R_{Kv}}^2 - d^2} \tag{4.19}$$

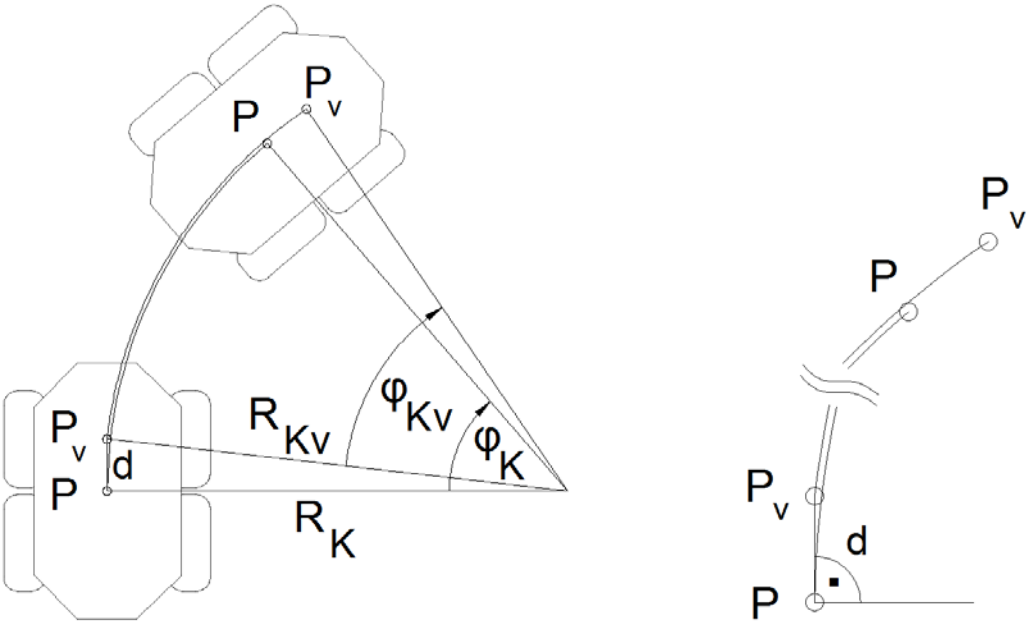


Abb. 4.4: Kreisradien bei unterschiedlicher Prismenposition und Detailansicht

Der bei der Kurvenfahrt überstrichene Winkel φ_K ist gleich groß wie φ_{Kv}, kann aus dem Verhältnis zwischen Bogenlänge und Radius errechnet werden und liefert zugleich die Änderung der Orientierung des Roboters.

$$\varphi_K = \frac{b_l}{R_K + L} = \frac{b_r}{R_K - L} \tag{4.20}$$

Mit Gleichung 4.10, Gleichung 4.11 und Gleichung 4.18 folgt:

$$\varphi_K = \frac{v_T \cdot t + \varphi_D \cdot r_D}{R_K + L} = \frac{v_T \cdot t - \varphi_D \cdot r_D}{R_K - L} \tag{4.21}$$

$$\varphi_K = \frac{v_T \cdot t + \omega_D \cdot r_D \cdot t}{L \cdot \frac{v_T}{\omega_D \cdot r_D} + L} = \frac{t \cdot (v_T + \omega_D \cdot r_D)}{\frac{L \cdot (v_T + \omega_D \cdot r_D)}{\omega_D \cdot r_D}} \tag{4.22}$$

Dieser Term liefert also

$$\varphi_K = \frac{\omega_D \cdot t \cdot r_D}{L} = \frac{\varphi_D \cdot r_D}{L} \tag{4.23}$$

und liefert den Zusammenhang zwischen der eingestellten Rotationsgeschwindigkeit ω_D und dem bei der Kurvenfahrt überstrichenen Winkel φ_K, welcher im nachfolgenden Abschnitt benötigt wird.

4.3 Transformation

Um die Positionsänderungen des Roboters bzw. des Prismas in einem übergeordneten Koordinatensystem beschreiben zu können, wird eine ebene Ähnlichkeitstransformation angewandt. Dazu wird ein lokales, rechtwinkeliges Roboter-Koordinatensystem x, y definiert, dessen Ursprung im Drehpunkt des Roboters liegt und dessen Ordinatenachse y mit der Längsachse des Roboters zusammenfällt. Wie bereits eingangs im Kapitel 4 erwähnt, wird die Höhenkomponente nicht berücksichtigt, weshalb auf eine für den dreidimensionalen Fall notwendige räumliche Ähnlichkeitstransformation mit sieben Parametern (drei Translationen, drei Rotationen sowie einem Maßstabsfaktor) verzichtet werden kann.

Das übergeordnete Koordinatensystem x^g, y^g entspricht in unserem Fall dem lokalen Koordinatensystem aus Abschnitt 6.1 und ist wie in Abbildung 4.5 zu erkennen gegenüber dem Roboter-Koordinatensystem um den Winkel θ verdreht. Die beiden durch x, y bzw. x^g, y^g aufgespannten Ebenen sind Parallel-Ebenen, der Normalabstand ergibt sich aus dem Höhenunterschied zwischen Boden und Mitte des Prismas. Eventuell auftretende Abweichungen durch die Unebenheit des Fliesenbodens sind minimal und können ebenso vernachlässigt werden wie das leichte Schwingen des Nick-Winkels (um die x-Achse) beim abrupten Anfahren des Roboters.

Das lokale Koordinatensystem x^g, y^g und das Roboter-Koordinatensystem x, y werden so wie alle weiteren verwendeten Koordinatensysteme durchgehend als mathematische Koordinatensysteme definiert.

Die Berechnung der Koordinatenunterschiede und Orientierungsänderungen erfordert eine Unterscheidung, wo das Prisma am Roboter befestigt ist. Im Folgenden werden die beiden Varianten „Prisma in der Mitte" und „Prisma vorne" daher getrennt voneinander betrachtet.

4.3.1 Variante 1: Prisma in der Mitte

Bei der ersten Variante in Abbildung 4.5 ist die Prismaposition P ident mit dem Drehpunkt des Roboters:

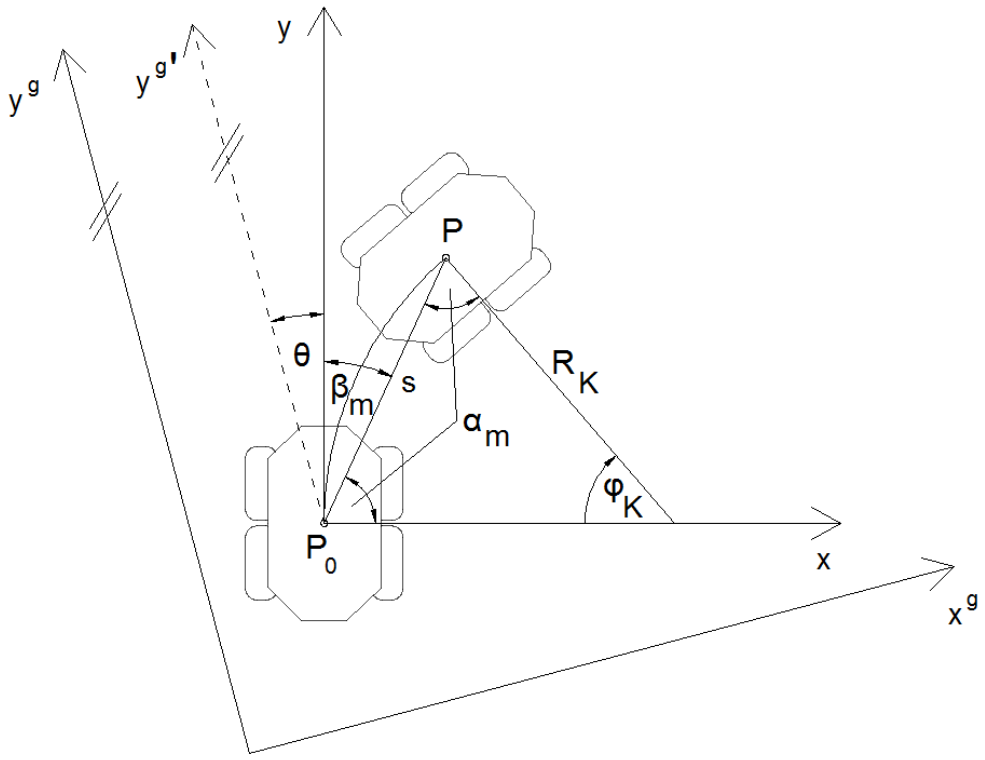


Abb. 4.5: Transformation in das übergeordnete Gangkoordinatensystem x^g, y^g; Prisma über Drehpunkt

Daraus lassen sich die beiden Winkel α_m und β_m (unter der Anwendung des Sehnentangentenwinkelsatzes) ableiten:

$$\alpha_m = \frac{\pi - \varphi_K}{2} \tag{4.24}$$

$$\beta_m = \frac{\varphi_K}{2} \tag{4.25}$$

$$s = 2 \cdot R_K \cdot \sin(\frac{\varphi_K}{2}) \tag{4.26}$$

Die neuen Koordinaten des Prismas P im übergeordneten Koordinatensystem x^g, y^g lauten daher:

$$x_P^g = x_{P_0}^g + s \cdot \sin(\theta + \beta_m) \tag{4.27}$$

$$y_P^g = y_{P_0}^g + s \cdot \cos(\theta + \beta_m) \tag{4.28}$$

Die neue Orientierung des Roboters ergibt sich aus der Summe des ursprünglichen Orientierungswinkels θ und des Winkels φ_K:

$$\theta_P^g = \theta + \varphi_K \tag{4.29}$$

Ersetzt man R_K nach Gleichung 4.18 und β_m nach Gleichung 4.25 erhält man:

$$x_P^g = x_{P_0}^g + 2 \cdot L \cdot \frac{v_T}{\omega_D \cdot r_D} \cdot \sin(\frac{\varphi_K}{2}) \cdot \sin(\theta + \frac{\varphi_K}{2}) \tag{4.30}$$

$$y_P^g = y_{P_0}^g + 2 \cdot L \cdot \frac{v_T}{\omega_D \cdot r_D} \cdot \sin(\frac{\varphi_K}{2}) \cdot \cos(\theta + \frac{\varphi_K}{2}) \tag{4.31}$$

Mit Gleichung 4.23 folgt schließlich, wobei gilt: $\varphi_D = \omega_D \cdot t$

$$x_P^g = x_{P_0}^g + 2 \cdot L \cdot \frac{v_T}{\omega_D \cdot r_D} \cdot \sin(\frac{\varphi_D \cdot r_D}{2 \cdot L}) \cdot \sin(\theta + \frac{\varphi_D \cdot r_D}{2 \cdot L}) \tag{4.32}$$

$$y_P^g = y_{P_0}^g + 2 \cdot L \cdot \frac{v_T}{\omega_D \cdot r_D} \cdot \sin(\frac{\varphi_D \cdot r_D}{2 \cdot L}) \cdot \cos(\theta + \frac{\varphi_D \cdot r_D}{2 \cdot L}) \tag{4.33}$$

$$\theta_P^g = \theta + \frac{\varphi_D \cdot r_D}{L} \tag{4.34}$$

4.3.2 Variante 2: Prisma vorne

Bei dieser Anordnung ist das Prisma P_v wie in Abbildung 4.6 dargestellt vom Mittelpunkt aus um den Abstand d versetzt über der Längsachse des Roboters angebracht.

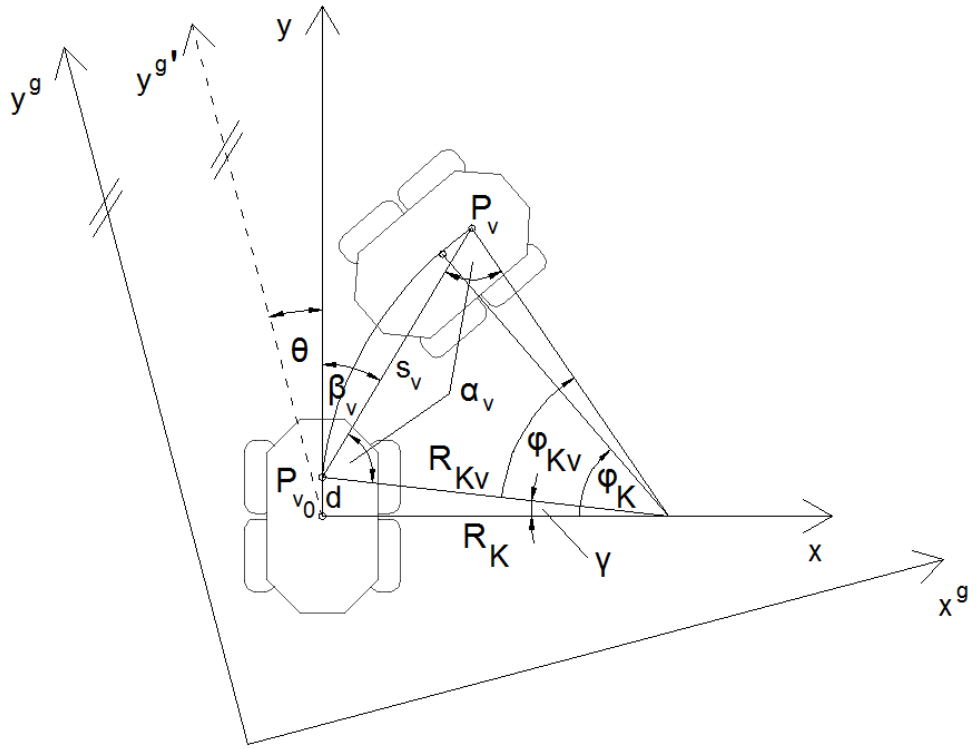


Abb. 4.6: Transformation in das übergeordnete Koordinatensystem x^g, y^g; Prisma über Vorderachse

Die Winkel α_v und β_v können aus dem vorigen Abschnitt von α_m und β_m direkt übernommen werden, ebenso bleibt $\varphi_{Kv} = \varphi_K$. Die restlichen Größen ergeben sich aus:

$$\gamma = \arctan(\frac{d}{R_K}) \tag{4.35}$$

$$\beta_v = \beta_m + \gamma \tag{4.36}$$

$$R_{Kv} = \sqrt{R_K^2 + d^2} \tag{4.37}$$

$$s_v = 2 \cdot R_{Kv} \cdot \sin(\frac{\varphi_{Kv}}{2}) \tag{4.38}$$

Die neuen Koordinaten des über der Vorderachse montierten Prismas P_v lauten daher im Koordinatensystem x^g, y^g:

$$x^g_{Pv} = x^g_{Pv_0} + s_v \cdot \sin(\theta + \beta_v) \tag{4.39}$$

$$y^g_{Pv} = y^g_{Pv_0} + s_v \cdot \cos(\theta + \beta_v) \tag{4.40}$$

$$\theta^g_{Pv} = \theta + \varphi_{Kv} \tag{4.41}$$

Analog zu Gleichung 4.32, Gleichung 4.33 und Gleichung 4.34 kann also durch Einsetzen der bekannten geometrischen Robotergrößen (Abstand Drehpunkt zum Mittelpunkt der vier Räder r_D, halbe Spurweite L) und der Einstellgrößen (Translationsgeschwindigkeit v_T, Rotationsgeschwindigkeit ω_D, Ausführungszeit t) die Änderung der Position und der Orientierung des Roboters berechnet werden.

5 Regelungssoftware

Die am Notebook laufende und in MATLAB implementierte Regelungssoftware enthält als Hauptteil den Regelkreis zur Steuerung des Roboters über die Soll-Trajektorie und übernimmt neben dem Empfang der Tachymeter-Trackingdaten über die GeoCOM-Schnittstelle auch die Rolle des Clients bei der TCP/IP-Socketverbindung mit dem Robotercomputer.

Zu Beginn wird die Soll-Trajektorie bestehend aus Geraden und Kreisbögen definiert und die Koordinaten der Start-, End- und Mittelpunkte werden wie in Abschnitt 6.2 beschrieben als Vektoren gespeichert. Danach werden die Verbindung zur Totalstation über die GeoCOM-Schnittstelle initialisiert (siehe Abschnitt 6.3), die für das Tracking notwendigen Einstellungen wie Prismentyp und Distanzmessprogramm an der Totalstation gesetzt und der Lock-Modus zur automatischen Zielverfolgung des 360°-Prismas aktiviert. Erst nach dieser bis zu zehn Sekunden dauernden Sequenz wird die TCP/IP-Socketverbindung zum Roboter-Serverprogramm aufgebaut. Dieses muss bereits vor dem Start der MATLAB-Routine ausgeführt werden und wartet auf die Verbindungsanfrage des Clients. Zur leichteren Bedienung des roboterinternen Computers während der Testmessungen wird vom Notebook mit dem Programm *PuTTY* eine *Secure Shell* (SSH) Verbindung über WLAN zum Rechner des Roboters aufgebaut. Durch diese verschlüsselte Netzwerkverbindung können in der Konsole eingegebene Befehle direkt am entfernten Rechner ausgeführt und somit auch das C++-Serverprogramm gestartet werden.

Bei erfolgreichem Verbindungsaufbau zum Tachymeter und zum Robotercomputer wird das Tracking des Prismas und somit der eigentliche Regelkreis gestartet.

5.1 Regelkreis

Der Regelkreis kann aus der in Kapitel 3 gezeigten Grundstruktur abgeleitet werden und ist in Abbildung 5.1 dargestellt. Die Führungsgröße $w(l)$ wird durch die Bedingung festgelegt,

dass die laterale Abweichung $q(t)$ der Roboterposition zur Soll-Trajektorie gleich Null ist. Durch die koordinative Vorgabe der Soll-Trajektorie kann zu jedem von der Totalstation gemessenen Prismenkoordinatenpaar die aktuelle Regelabweichung $e(t) = q(t)$ zur Soll-Trajektorie bestimmt werden. Die Regeleinrichtung ist in dieser Arbeit aus Zeitgründen nur als proportionaler Regler konzipiert und berechnet aufgrund der Regelabweichung die Rotationsgeschwindigkeit $\omega(t)$ des Roboters als Stellgröße $u(t)$. Für die in Kapitel 7 durchgeführten Testfahrten wurde als Proportionalitätsfaktor $K_p = 50$ empirisch gewählt, der zufällig mit der Translationsgeschwindigkeit v_T in der Einheit mm/sec übereinstimmt. Der Roboter repräsentiert durch seine Bewegung die Regelstrecke, wobei Störeinflüsse wie zum Beispiel die Beschaffenheit des Untergrunds auf ihn einwirken können. Sein Stellverhalten kann auch durch roboterinterne Störeinflüsse beeinträchtigt werden wie etwa unterschiedlicher Reifendruck oder durch die Antriebsriemen. Die Totalstation als Messglied liefert die aktuelle Ist-Position des 360°-Prismas, welche durch die mangelnde Synchronisation der Sensordaten und Abweichungen durch die Verwendung des 360°-Prismas fehlerbehaftet ist.

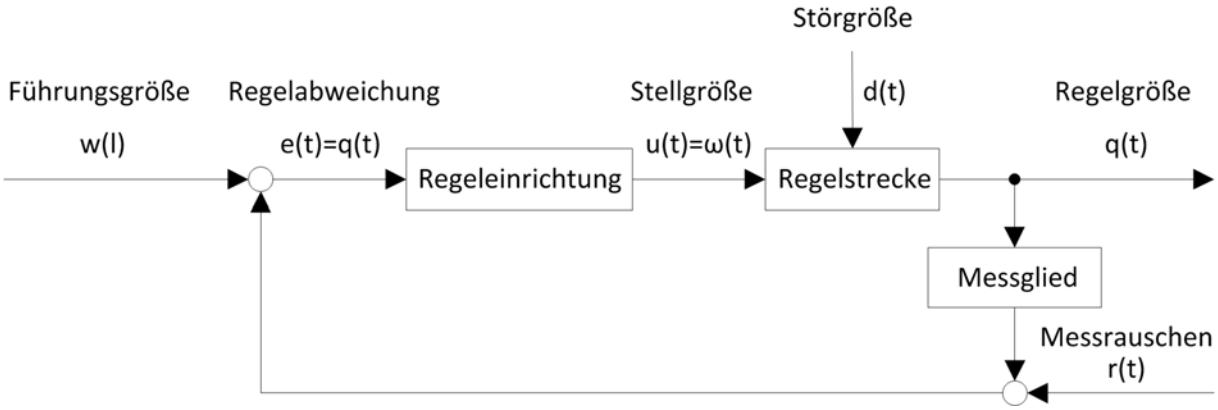


Abb. 5.1: Regelkreis zur Robotersteuerung entlang einer Soll-Trajektorie

Bei Geraden ist die Rotationsgeschwindigkeit standardmäßig Null und wird auch nicht geändert, solange sich die Prismenposition innerhalb des Toleranzbandes von ±1 cm bewegt, erst darüber hinaus wird eine Gegensteuerung eingeleitet. Bei geringen Translationsgeschwindigkeiten ist allerdings auch keine feinere Nachsteuerung möglich, da nur ganzzahlige Rotationsgeschwindigkeiten (°/sec) eingestellt werden können. Diese vom Roboter bedingte Einschränkung bewirkt, dass der proportionale Regler zwar stetige Stellgrößenwerte berechnet, diese aber nur in diskreten Werten von der Regelstrecke übernommen werden können. Zur Verdeutlichung ist in Abbildung 5.2 die Kennlinie der

Regeleinrichtung und der Regelstrecke dargestellt. Die Regeleinrichtung liefert außerhalb des Nullbereichs von ±1 cm Regelabweichung ($e(t)$) eine proportionale Stellgröße $u(t)$ als Ausgangsgröße. Der Roboter als Regelstrecke kann dagegen nur ganzzahlige Stellwerte für die Rotationsgeschwindigkeit verarbeiten, weshalb sich die Stufenregelung in der rechten Grafik ergibt.

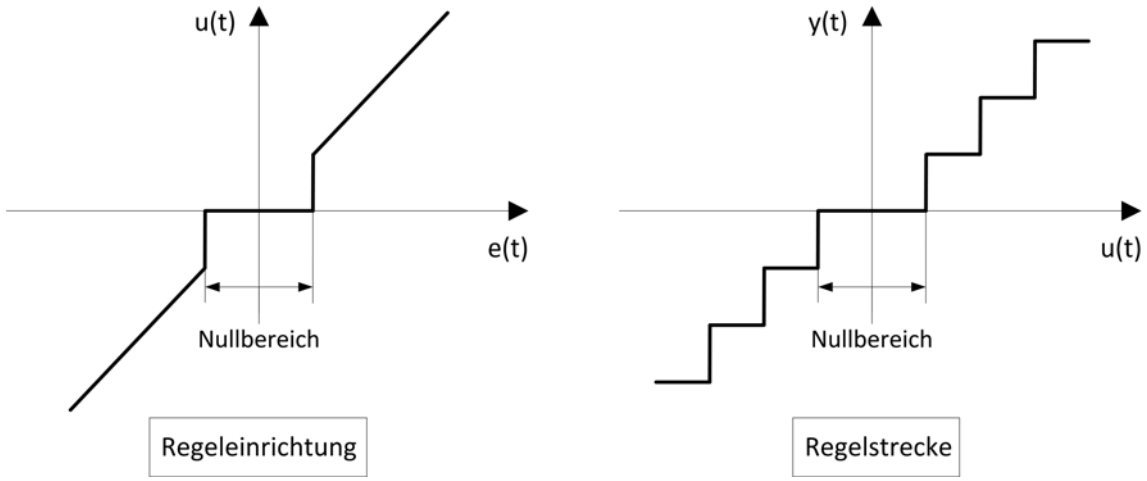


Abb. 5.2: Kennlinie der Regeleinrichtung und der Regelstrecke

Durch die Zusammenhänge zwischen dem Kurvenradius und der Rotationsgeschwindigkeit bei einer bestimmten Translationsgeschwindigkeit aus der durchgeführten Kalibrierung (siehe Abschnitt 7.3) wird eine Vorsteuerung beim Durchfahren von Kreisbögen angewendet: Zu Beginn des Bogens wird die vorausberechnete Drehgeschwindigkeit eingestellt und je nach gemessener Abweichung vom Regler korrigiert. Weitere fahrdynamische Eigenschaften und externe Störeinflüsse des Roboters auf den Regelkreis wurden nicht berücksichtigt.

5.2 Berechnung der Abweichung zur Soll-Trajektorie

Ein wesentlicher Bestandteil des Regelkreises zum Abfahren der Soll-Trajektorie besteht darin, die aktuelle laterale Abweichung $q(t)$ des am Roboter befestigten Prismas festzustellen. Dabei ist nicht nur die Größe, sondern auch die Richtung der Abweichung (in Fahrtrichtung gesehen links oder rechts) von Relevanz. Die Art der Berechnung dieser Parameter ist davon abhängig, ob es sich beim aktuellen Trajektorien-Element um eine Gerade oder einen Kreisbogen handelt. Daher werden die beiden Elemente in den folgendenen zwei Abschnitten getrennt voneinander betrachtet.

5.2.1 Gerade

Im Fall der Roboterbewegung entlang einer Geraden, die in Abbildung 5.3 durch den Anfangspunkt A und den Endpunkt B (Vektor $\overrightarrow{g}$) gegeben ist, soll die aktuelle Abweichung als Normalabstand $q(t)$ vom Prismenpunkt P zur Geraden $\overrightarrow{g}$ berechnet werden.

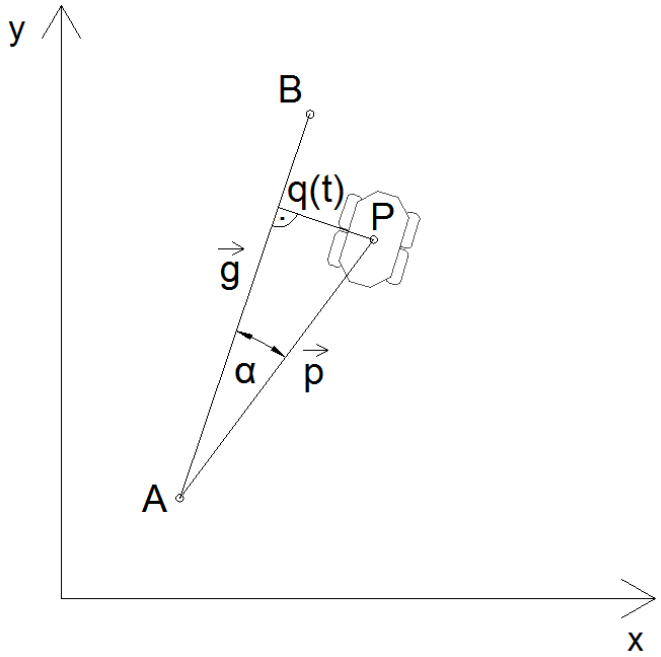


Abb. 5.3: Abweichung $q(t)$ des Punktes P zur Soll-Trajektorie (Gerade $\overrightarrow{g}$)

Der eingeschlossene Winkel α kann über das Skalarprodukt der beiden Vektoren $\overrightarrow{g}$ und $\overrightarrow{p}$ ermittelt werden:

$$cos(\alpha) = \frac{\overrightarrow{g} \cdot \overrightarrow{p}}{||\overrightarrow{g}|| \, ||\overrightarrow{p}||} \tag{5.1}$$

In weiterer Folge kann der Normalabstand $q(t)$ abgeleitet werden:

$$b = \overrightarrow{p} \cdot sin(\alpha) \tag{5.2}$$

Bestimmung der Richtung der Abweichung zur Geraden Die rechnerische Entscheidung, ob die eben bestimmte Abweichung $q(t)$ in Fahrtrichtung gesehen links oder rechts von der Geraden $\overrightarrow{g}$ liegt, wird mittels der Richtungswinkel der Vektoren $\overrightarrow{g}$ und $\overrightarrow{p}$ gelöst.

Dieser kann in MATLAB mit dem Befehl *atan2* berechnet werden und liefert Ergebnisse von $-180°$ bis $+180°$, wobei die 0°-Marke wie in Abbildung 5.4 dargestellt mit der Richtung der Ordinatenachse zusammenfällt.

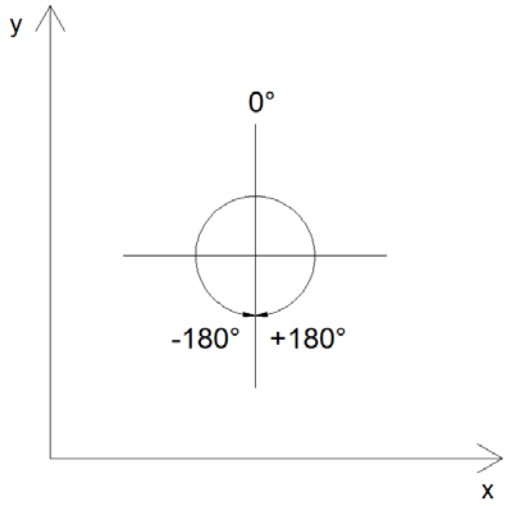


Abb. 5.4: Wertebereich der Orientierungsberechnung mittels der atan2-Funktion

Dabei soll zum Beispiel auch berücksichtigt werden, dass der Roboter zu Beginn einer Trajektorien-Gerade den Anfangspunkt erst passieren muss und daher noch nicht innerhalb des „Normalabstandbereichs" der beiden Geradenpunkte - also im Korridor der links und rechts verlängerten Normalvektoren auf Anfangs- und Endpunkt der Gerade - liegt. Diese Entscheidungsfindung kann mit einer Fallunterscheidung erreicht werden, ob der Richtungswinkel der Geraden positiv oder negativ ist:

Fall 1: Geraden-Orientierungswinkel positiv In diesem Fall nimmt der Orientierungswinkel der Geraden von A nach B einen Wert zwischen 0° und 180° ein. Ist der Orientierungswinkel des Abweichungsvektors größer als der der Geraden, so befindet sich der Roboter in Fahrtrichtung gesehen rechts neben der Soll-Strecke. In Abbildung 5.5 sind diese beiden Bereiche schraffiert dargestellt. Jede andere festgestellte Abweichung kann somit als „links" klassifiziert werden.

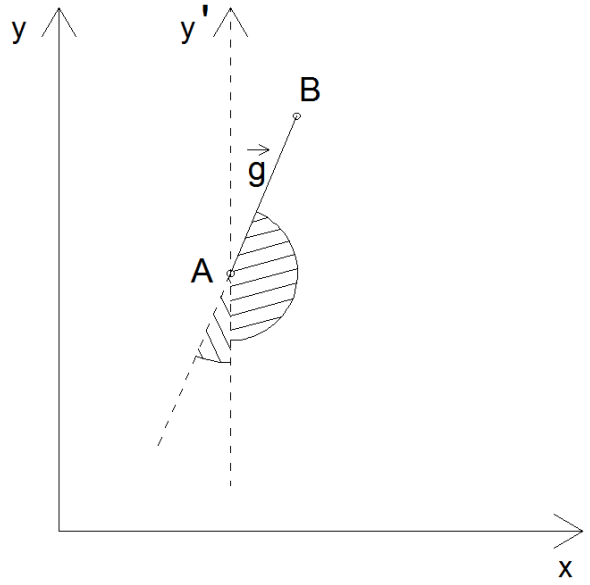


Abb. 5.5: Positiver Richtungswinkel der Geraden $\overrightarrow{g}$

Fall 2: Geraden-Orientierungswinkel negativ Bei einem negativen Richtungswinkel der Geraden $\overrightarrow{g}$ erfolgt die Abfrage der Abweichungsrichtung ähnlich: Der größere schraffierte Kreissektor in Abbildung 5.6 deckt den Bereich ab, wo die Orientierung des Abweichungsvektors $\overrightarrow{p}$ kleiner ist als die der Geraden; der kleinere schraffierte Kreissektor den Bereich, wo der Orientierungswinkel größer ist als die gegen die Geradenrichtung verlängerte Orientierung. Zusammen stellen die zwei Sektoren den Bereich dar, wo die Abweichung der aktuellen Roboterposition in Fahrtrichtung gesehen links neben der vorgegebenen Strecke liegt. Jeder andere Orientierungswinkel des Abweichungsvektors liegt daher rechts neben oder im Spezialfall exakt auf der Soll-Geraden.

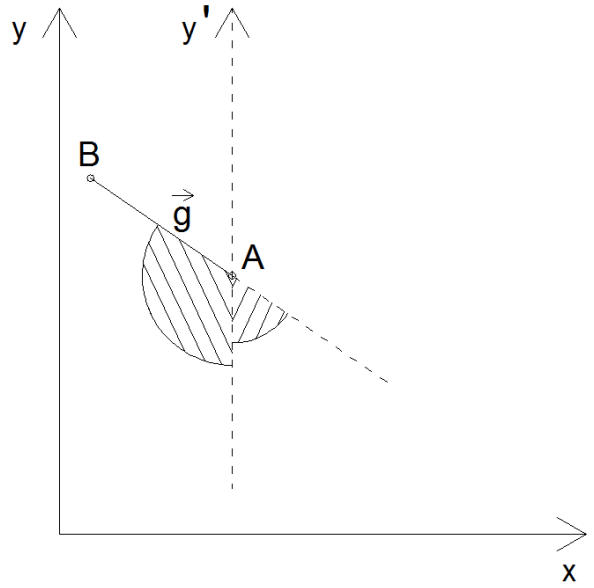


Abb. 5.6: Negativer Richtungswinkel der Geraden $\overrightarrow{g}$

5.2.2 Kreisbogen

Bewegt sich der Roboter entlang eines Kreisbogens so wie in Abbildung 5.7 dargestellt von Punkt A nach B, so dient die radiale Abweichung $q(t)$ vom Mittelpunkt M_K zur Prismenposition P als Abweichungsgröße:

$$q(t) = \overline{M_K P} - R_K \tag{5.3}$$

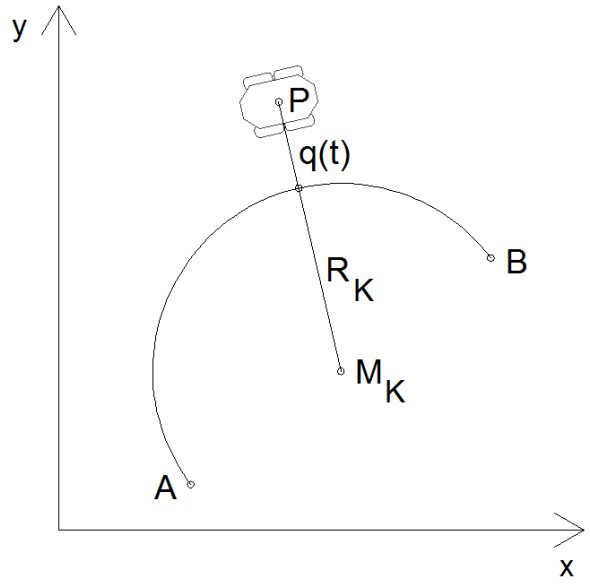


Abb. 5.7: Radiale Abweichung $q(t)$ des Punktes P zur Soll-Trajektorie (Kreisbogen mit Radius R_K)

Eine Abweichung $q(t)$ größer als Null bedeutet, dass sich der Roboter außerhalb des Kreisbogens befindet und zur Gegensteuerung die Rotationsgeschwindigkeit des Roboters weiter vergrößert werden muss. Fährt der Roboter innerhalb des Kreisbogens ist die Abweichung $q(t)$ kleiner als Null und erfordert daher eine Reduktion der Rotationsgeschwindigkeit.

5.3 Abbruchkriterium beim Übergang zwischen zwei Trajektorie-Elementen

Zum Abfahren der Soll-Trajektorie muss ein Abbruchkriterium definiert werden um zu erkennen, wann der Roboter ein Trajektorien-Element fertig abgefahren hat und auf welches Element sich die Abweichungsberechnungen der aktuellen Prismenkoordinaten beziehen sollen. Als Abbruchbedingung könnte man zum Beispiel festlegen, wenn sich die Position des Roboters in einem definierten Umkreis um den Endpunkt der Gerade oder des Kreisbogens befindet. Ist allerdings die Abweichung des Roboters am Ende des jeweiligen Streckenelements zu groß und fällt nicht in diese Endpunktumgebung, versagt das Abbruchkriterium. Daher wird, wie bei der Abweichungsberechnung in Abschnitt 5.2, auf eine vektorielle Berechnungsmethode zurückgegriffen und die Elemente Gerade und

Kreisbogen werden wieder getrennt voneinander betrachtet.

5.3.1 Abbruchkriterium Gerade

Das Kriterium, ob der Roboter noch entlang einer Gerade fährt oder diese bereits passiert hat, wird mit der Projektion des Vektors $\overrightarrow{p}$ auf den Geradenvektor $\overrightarrow{g}$ abgefragt und ist in Abbildung 5.8 grafisch dargestellt.

$$\overrightarrow{a} = \overrightarrow{p} \cdot cos(\alpha) \tag{5.4}$$

Ist $|\overrightarrow{a}|$ größer als $|\overrightarrow{g}|$, so ist das Abbruchkriterium erfüllt und die Regelabweichungen werden auf das darauffolgende Trajektorien-Element bezogen oder die Fahrt beendet. Durch die Abhängigkeit vom Kosinus des eingeschlossenen Winkels in der Abfrage kann es auch passieren, dass der Regelkreis unterbrochen wird, obwohl der Roboter den Kurs (mit dem ersten Element Gerade) noch gar nicht abgefahren ist. Dies ist der Fall, wenn der Abstand zwischen der Roboterposition und dem Anfangspunkt A größer ist als $|\overrightarrow{g}|$. Dieser Bereich ist in Abbildung 5.8 schraffiert dargestellt.

Doch auch innerhalb dieses Bereichs sowie neben der Geraden kann es passieren, dass der Regelkreis und somit die Annäherung des Roboters an die Sollstrecke versagt. Ist nämlich die Abweichung zur Geraden sehr groß, versucht der p-Regler dementsprechend mit einer höheren Rotationsgeschwindigkeit gegenzusteuern. Trifft der Roboter allerdings durch dieses Manöver nicht auf die Sollstrecke zurück, so dreht er sich im Kreis (vergleichbar mit Abbildung 7.21 rechts auf Seite 72) und der Regelungsalgorithmus versagt. Aus diesem Grund muss zu Beginn der Fahrt darauf geachtet werden, dass der Roboters hinreichend nahe am Startpunkt der Geraden positioniert wird.

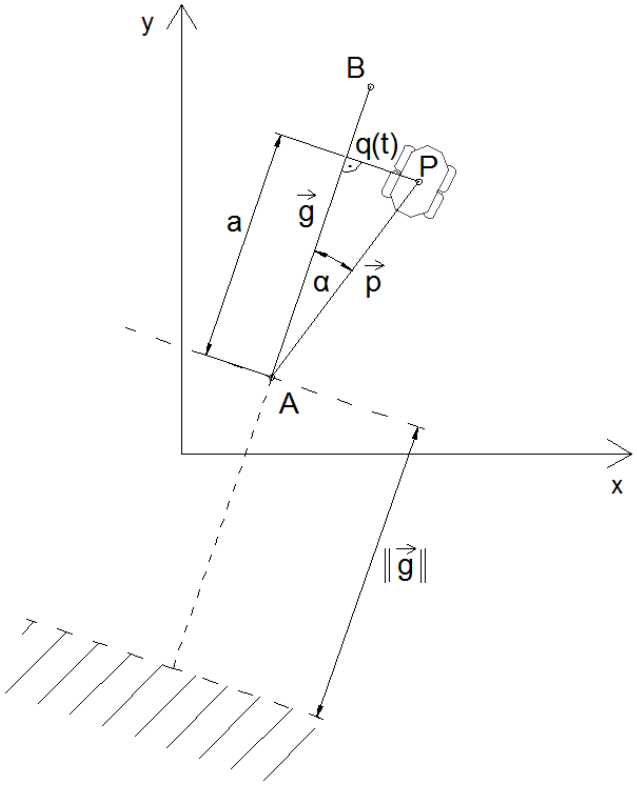


Abb. 5.8: Abbruchkriterium für Geraden

5.3.2 Abbruchkriterium Kreisbogen

In Tabelle 5.1 sind die Abbruchkriterien beim Durchfahren von Kreisbögen grafisch dargestellt. Die schraffierten Bereiche zeigen jeweils den Winkelbereich für Rechtsdrehungen, der durch den Kreisbogen vom Startpunkt A zum Endpunkt B überstrichen wird. Die sechs Variationsmöglichkeiten, ob Anfangs- bzw. Endorientierung des Kreisbogens positiv oder negativ sind, können auf zwei Fälle aufgeteilt werden, nämlich ob der Richtungswinkel t_A vom Startpunkt größer oder kleiner ist als der Richtungswinkel t_B zum Endpunkt.

Gelten die Bedingungen

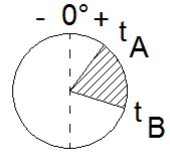
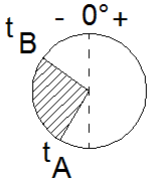
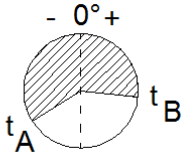
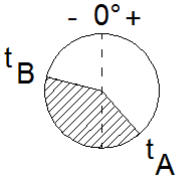
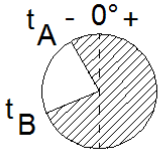
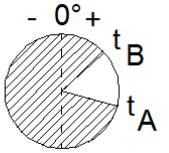
$$t_A < t_B \text{ und } t_A \leq t_{MP} < t_B,$$

so durchfährt der Roboter noch den Kreisbogen, wobei t_{MP} der Richtungswinkel der momentanen Prismenposition ist.

Ist der Richtungswinkel t_A größer als t_B, befindet sich der Roboter noch auf dem Kreisbogen, wenn gilt:

$$t_{MP} \geq t_A \text{ oder } t_{MP} < t_B.$$

Der Sonderfall der gleichen Orientierung beider Kreisbogenpunkte wird dadurch vermieden, dass eine volle Kreisbewegung durch zwei Halbkreise definiert wird. Für die Abfragen der Linksdrehung müssen diese logisch verneint werden.

Grafische Darstellung			Abfragebedingung
			Fall 1: $t_A < t_B$ $t_A \leq t_{MP} < t_B$
			Fall 2: $t_A > t_B$ $t_{MP} \geq t_A$ oder $t_{MP} < t_B$

Tab. 5.1: Abbruchkriterien für Kreisbögen (Rechtsdrehung)

6 Versuchsaufbau und vorbereitende Arbeiten

Als Versuchsort wurde der Gang zwischen Treppenhaus und Lift im dritten Stockwerk des Institutsgebäudes Gußhausstraße 27-29 (Neues EI) der Technischen Universität Wien gewählt. Die erste Überlegung, die Tests am Messdach durchzuführen, wurde aufgrund mehrerer Argumente verworfen. Messungen am Dach wären durch die nur teilweise Überdachung stark wetterabhängig gewesen. Weiters konnten durch die in Kies gelegten Waschbetonplatten keine gleichmäßigen Bewegungen des Roboters erwartet werden und der ständige Transport der gesamten Messausrüstung (Roboter, Stativ, Tachymeter, Notebook,...) wäre sehr aufwändig gewesen. Den Aufbau der gesamten Ausrüstung am Gang zeigt Abbildung 6.1.

Abb. 6.1: Versuchsaufbau mit gesamter Messausrüstung

Für den Teil der praktischen Erprobung, die Steuerung des Roboters entlang einer vorgegebenen Strecke, waren zahlreiche Vorarbeiten notwendig, auf die in den folgenden Unterkapiteln näher eingegangen wird.

6.1 Lokales Koordinatensystem

Als Grundlage für die Testmessungen wurde am Gang ein lokales Koordinatensystem definiert und eingemessen. Dafür wurden fünf selbstklebende Reflektorfolien (20x20 mm) der Firma Leica Geosystems als Fixpunkte verwendet, die in bestmöglicher geometrischer Anordnung an Türrahmen, Säulen und Wänden befestigt wurden.

Das Koordinatensystem wurde bei der anschließenden CAD-Auswertung so ausgerichtet, dass die Koordinatenachsen näherungsweise mit den Fugen des Fliesenbodens zusammenfallen. Weiters wurde darauf geachtet, dass der Koordinatenursprung in eine Ecke der verfügbaren Fläche gelegt wurde und somit nur positive Koordinatenwerte behandelt werden müssen. Diese spezielle Lagerung des Systems erleichtert auch die eigene Vorstellung und die Orientierung im Raum. Die Positionen der reflektierenden Zielmarken $P1$ bis $P5$ und der Ursprung des Systems sind im Übersichtsplan in Abbildung 6.2 abgebildet.

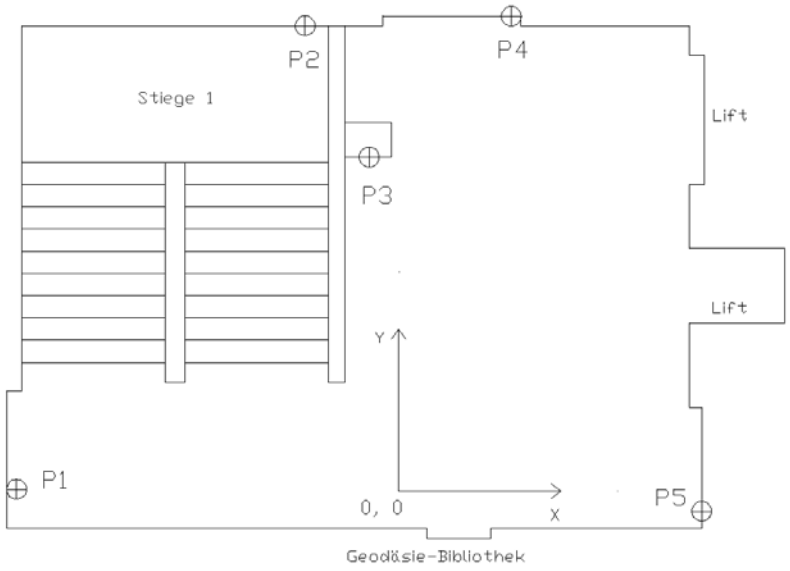


Abb. 6.2: Lokales Koordinatensystem im 3. Stock des Neuen EI der TU Wien, Gußhausstraße 27-29

6.2 Definition Soll-Trajektorie

Bei der Vorgabe der Soll-Trajektorie werden der Einfachheit halber nur die Elemente *Gerade* und *Kreisbogen* verwendet. Während sich die Definition einer Gerade auf die Angabe von Start- und Endpunkt beschränkt, sind beim Kreisbogen zusätzlich noch der Mittelpunkt und die Drehrichtung anzugeben. Die zweidimensionalen Punktkoordinaten repräsentieren dabei bereits die Lage der Trajektorie im lokalen Koordinatensystem. Durch eine freie Stationierung der Totalstation in diesem Koordinatensystem können die gemessenen Prismenkoordinaten direkt mit den Koordinaten der Soll-Trajektorie verglichen werden.

Als Beispiel einer Soll-Trajektorie ist in Abbildung 6.3 eine ovale Strecke bestehend aus zwei Geraden und zwei Halbkreisbögen dargestellt, die in den praktischen Tests hauptsächlich als Soll-Strecke verwendet wurde.

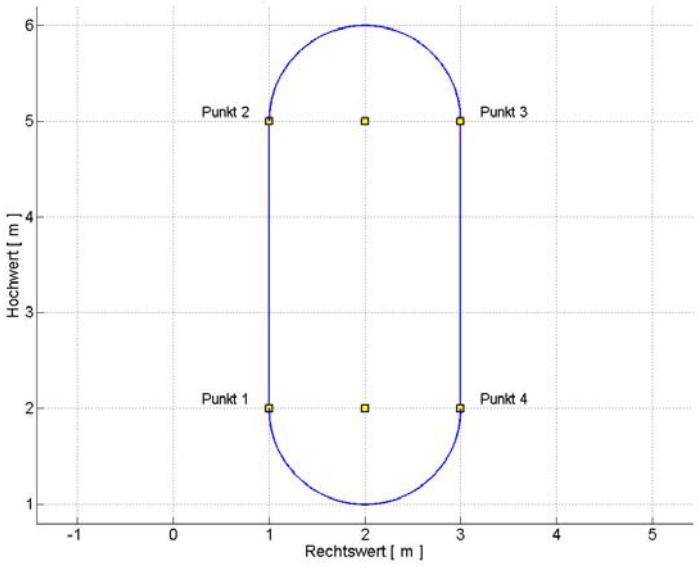


Abb. 6.3: Vorgabe einer Soll-Trajektorie mit zwei Geraden und 2 Kreis-Halbbögen im lokalen Gang-Koordinatensystem

In MATLAB werden die Elemente jeweils als Vektor gespeichert, die Definition des Kreisbogens von Punkt 2 nach Punkt 3 ist in Tabelle 6.1 dargestellt. Die redundante Definition des Kreisbogens durch die Angabe von sieben anstatt von sechs nötigen Parametern wurde gewählt, um sonst weitere notwendige Berechnungen im MATLAB-Programm zu

vermeiden.

Wert	Beschreibung
1	Elementtyp (0 = Gerade, 1 = Kreisbogen)
1.000	Rechtswert Startpunkt [m]
5.000	Hochwert Startpunkt [m]
3.000	Rechtswert Endpunkt [m]
5.000	Hochwert Endpunkt [m]
2.000	Rechtswert Mittelpunkt [m]
5.000	Hochwert Mittelpunkt [m]
2	Drehrichtung (1 = links, 2 = rechts)

Tab. 6.1: Definition des Kreisbogens von Punkt 2 nach Punkt 3 der Soll-Trajektorie aus Abbildung 6.3

6.3 Tracking

Die Aufgaben des Trackings mit der Totalstation liegen einerseits bei der Positionsbestimmung des 360°-Prismas in Echtzeit als Grundlage für den Roboter-Steuerungsprozess. Andererseits dient die permanente Aufzeichnung der Prismenkoordinaten gleichzeitig auch dem Zweck der Dokumentation der Ergebnisse und zur Darstellung der Abweichungen zur Soll-Trajektorie. Eine unabhängige Erfassung der Abweichungen ist daher durch diese Methode nicht gewährleistet und sollte in einer weiterführenden Arbeit durch ein unabhängiges Messverfahren berücksichtigt werden.

Dazu wird das Tachymeter des Typs TCRP 1201+ von Leica Geosystems im sogenannten Lock-Modus betrieben, der eine permanente Zielverfolgung des 360°-Prismas ermöglicht. Die Ausführung der elektronischen Distanzmessung (EDM) im Trackingmodus erlaubt so eine Messrate für 3D-Punktkoordinaten von etwa 10 Hertz. Das bedeutet, dass eine Distanzmessung im Trackingmodus in 0,1 Sekunden abgeschlossen sein muss, während bei einer Standardmessung eine Genauigkeitssteigerung durch viele Einzelmessungen in einem Zeitraum von etwa 1,5 Sekunden erreicht wird. Im Vergleich zur Distanzmessung kann der Abgriff der Horizontalrichtung und des Vertikalwinkels viel schneller erfolgen. Der Umstand der unterschiedlichen Messzeiten führt allerdings zu einer fehlenden zeitlichen Synchronisation der einzelnen polaren Messelemente und kann in Abhängigkeit von der Objektgeschwindigkeit und der geometrischen Situation einen Punktkoordinatenfehler von

bis zu mehreren Dezimetern verursachen (Stempfhuber et al., 2000; Stempfhuber und Maurer, 2001).

Die Kommunikation zwischen Notebook und Tachymeter wird über das von Leica Geosystems bereitgestellte GeoCOM-Protokoll realisiert, wobei die Verbindung über die serielle Schnittstelle hergestellt wird. Als Einstellungen müssen die Parameter der Schnittstelle wie die Com-Portnummer, Baudrate, Parität, Protokoll, Daten- und Stoppbits sowohl am Notebook als auch an der Totalstation festgelegt werden.

Das Senden von GeoCOM-Befehlen an die Totalstation und das Auslesen von Messdaten erfolgt über das ASCII-Protokoll, wobei auf jede Anfrage des Computers eine Antwort des Tachymeters folgt. Die Tachymeterfunktion einer einfachen Distanzmessung lautet zum Beispiel *TMC_QuickDist* und wird mit folgendem Befehl (ASCII-Request) ausgelöst:

$$'R1Q, 2117 :'$$

Die Antwort der Totalstation (ASCII-Response) wird in der Form

$$'R1P, 0, 0 : RC, dHz[double], dV[double], dSlopeDistance[double]'$$

zurückgegeben und liefert bei erfolgreicher Ausführung den Returncode (RC) 0 und als Messwerte den Horizontalwinkel (*dHz*), den Vertikalwinkel (dV) und die Schrägdistanz (*dSlopeDistance*). Konnte keine Distanz gemessen werden, enthält der Returncode den entsprechenden Fehlercode und es werden nur Horizontal- und Vertikalwinkel zurückgegeben.

Eine detailliertere Beschreibung aller GeoCOM-Funktionen kann dem *GeoCOM Reference Manual* von Leica Geosystems entnommen werden.

6.3.1 Fehlereinfluss durch Verwendung eines 360°-Prismas zur automatischen Zielverfolgung

Neben den bereits zu Beginn von Abschnitt 6.3 beschriebenen Genauigkeitsverlusten bei kinematischen Messungen durch die mangelhafte Synchronisation der verschiedenen Messsensoren bringt auch die Verwendung eines 360°-Prismas eine weitere Fehlerquelle mit sich. Solche Rundum-Prismen sind aus mehreren Tripelprismen aufgebaut und bieten zwar

den Vorteil, dass sie unabhängig von deren horizontalen Ausrichtung zum Tachymeter angezielt werden können. Dem gegenüber steht jedoch der Nachteil, dass der prismenabhängige Fehler bis zu mehreren Millimetern betragen kann (Hennes, 2000; Stempfhuber und Kirschner, 2008). Dieser Fehleranteil entsteht durch Brechung der EDM- und ATR-Sensorstrahlung am Glas des Reflektors (Ingensand, 2001).

In Abbildung 6.4 ist die Anzielung eines 360°-Prismas und das jeweilige Spotbild der ATR-Ablagenmessung gegenübergestellt. Im linken Teil der Abbildung ist der Zielstrahl optimal auf die Prismenfläche ausgerichtet und erzeugt ein Spotbild, das mit dem eines Präzisionsreflektors verglichen werden kann. Erfolgt jedoch eine horizontale Prismenverdrehung und somit die Messung auf den Kantenübergang, so entstehen zwei in entgegengesetzter Richtung diagonal verschobene Spotbilder (siehe Abbildung 6.4 rechts oben).

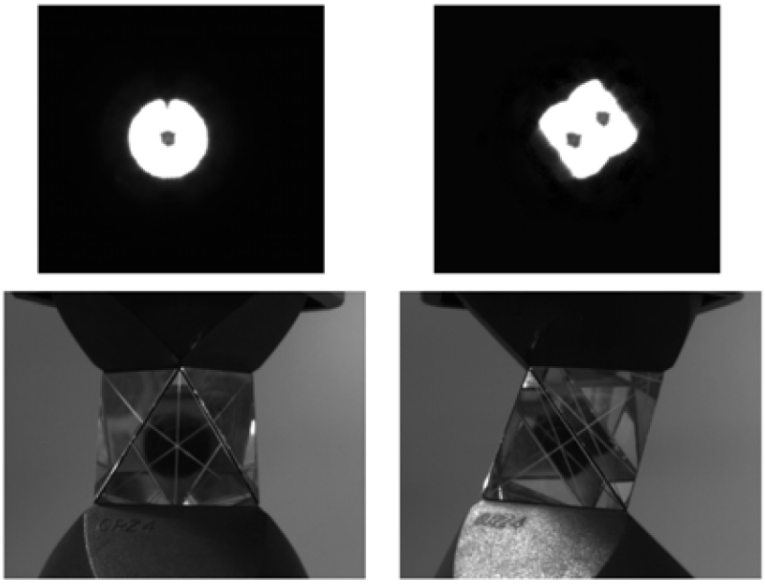


Abb. 6.4: Anzielung eines 360°-Prismas (links: frontal auf eine Prismenfläche; rechts: horizontal verdreht) und darüber die daraus resultierenden Spotbilder (Stempfhuber und Kirschner, 2008)

Die Auswirkung auf die Längs- und Querabweichungen während einer vollen 360°-Drehung des Prismas zeigt Abbildung 6.5. Die periodischen Abweichungsschwankungen ergeben sich aus den bereits beschriebenen Messungen bei den Kantenübergängen und sind in Querrichtung mit etwa -3 mm bis +3 mm am größten.

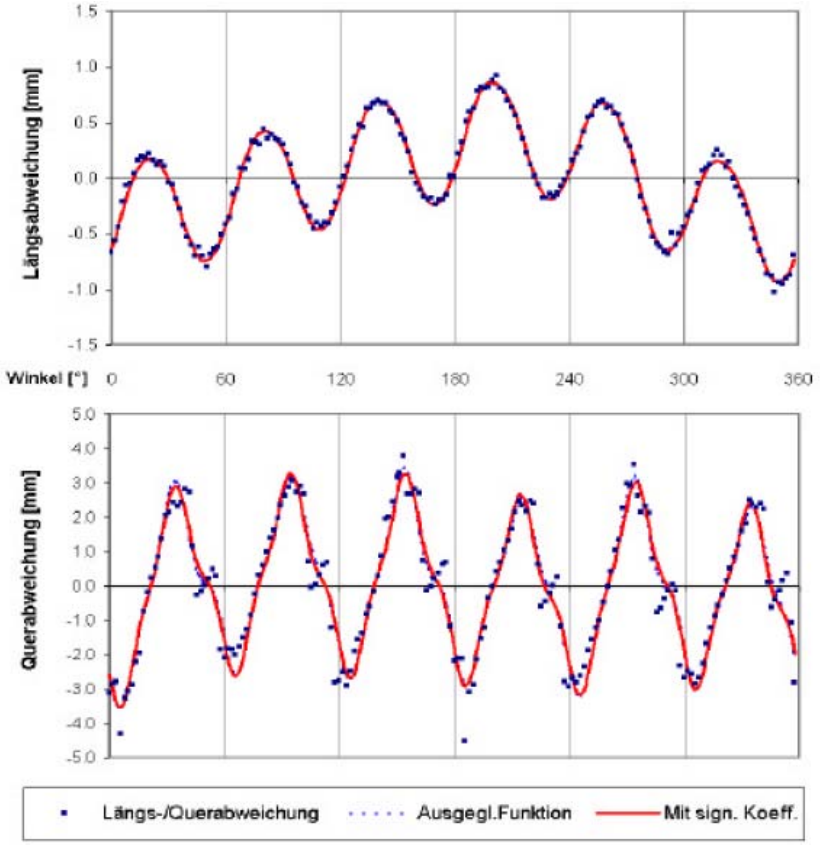


Abb. 6.5: Abweichungen in Längs- und Querrichtung während einer vollen Umdrehung des 360°-Prismas (Hennes und Favre, 2000)

Die möglichen Fehlereinflüsse durch die Verwendung eines 360°-Prismas auf diese Arbeit werden in Abschnitt 7.2 untersucht.

6.4 TCP/IP Socketverbindung Notebook / Seekur Jr

Die Kommunikation zwischen Notebook und Roboter wird über eine TCP/IP Socketverbindung realisiert. Durch den am Roboter montierten WLAN-Router kann diese Socketverbindung über das *Wireless Local Area Network* (WLAN) aufgebaut werden und es ist daher auch während der Testfahrten keine Kabelverbindung zum Notebook erforderlich.

Die Socketprogrammierung ist die Grundlage bei der Entwicklung von verteilten Anwendungen in Client-Server-Architekturen, wobei in diesem Fall das Notebook als Client und der Roboter als Server fungiert. Ein Socket (englisch für Steckdose) kann mit einer

Datei verglichen werden: nach dem Öffnen kann er von einem Programm beschrieben und gelesen werden und muss am Ende wieder geschlossen werden. Der Server bindet sich beim Erstellen eines Sockets an einen fixen Port, über den er erreichbar ist und wartet auf Anfragen von einem oder mehreren Clients. Dem Client muss für den Verbindungsaufbau der Port des Servers bekannt sein, um danach Anfragen an den Server zu schicken und Antworten zu erhalten. Die Plattformunabhängigkeit von Sockets erlaubt es im Fall der Robotersteuerung, dass am Roboter ein C++ Serverprogramm unter Linux mit einem MATLAB-Clientprogramm unter Windows kommunizieren kann.

6.5 Serverprogramm Seekur Jr

Durch die Bereitstellung zahlreicher komplexer Klassenbibliotheken in der Programmiersprache C++ durch die Herstellerfirma MobileRobots wurde auch das Serverprogramm für die TCP/IP Socketverbindung in C++ implementiert.

Das bereits in Kapitel 2 beschriebene Problem der unterschiedlich langen Ausführungsdauer gleicher Befehlssequenzen führte dazu, dass die Roboterbewegung ruckhaft erfolgte, wenn die Befehle vom Notebook abgeschickt wurden. Da eine solche unregelmäßige Fahrweise nicht erwünscht ist, wurde das anfangs geplante Konzept der Einzelbefehle verworfen. Stattdessen soll sich der Roboter mit einer gleichmäßigen Geschwindigkeit vorwärts bewegen und auf Abweichungen von der Soll-Trajektorie mit einer Änderungen der Rotationsgeschwindigkeit reagieren. Dieser Umstieg bei der Befehlsausführung bedingt auch eine Änderung der Client-Server-Kommunikation, vor allem aber beim Serverprogramm.

Die angestrebte konstante Fahrbewegung wird durch die Verwendung der Multithreading-Technologie am Server erreicht: Ein Thread kümmert sich um die eintreffenden Clientanfragen, während der zweite Thread für die gleichmäßige Fahrbewegung des Roboters und die Reaktion auf Befehlsänderungen zuständig ist. Abbildung 6.6 stellt die Kommunikation zwischen Client und Server und die Threadingfunktionalität des Servers in vereinfachter schematischer Form dar.

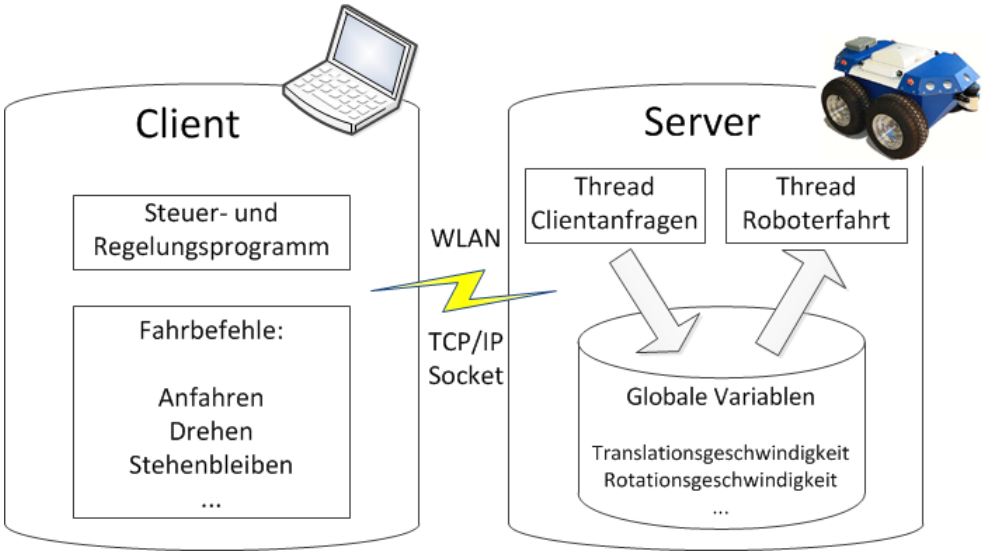


Abb. 6.6: Schematische Darstellung der Kommunikation zwischen Client und Server

Durch die (scheinbar) parallele Ausführung beider Threads können Steuerbefehle des Clients sehr schnell umgesetzt werden, indem Parameter, wie zum Beispiel die Rotationsgeschwindigkeit, abgeändert werden. Diese Parameter sind im Serverprogramm als globale Variablen realisiert und werden vom für die Clientanfragen zuständigen Thread geändert. Die Threads werden deshalb nur scheinbar parallel ausgeführt, weil für ein „echtes" Multithreading ein Mehrprozessorsystem nötig wäre. Ist, wie beim Computer des Roboters, nur ein Prozessor vorhanden, wechselt das Betriebssystem regelmäßig zwischen den Threads, was als *time-splicing* bezeichnet wird.

Für die eventuelle spätere Verwendung der Odometerdaten des Roboters im Steuer- und Regelungsprogramm wurde auch eine Möglichkeit implementiert, dass der Client in regelmäßigen zeitlichen Abständen die Odometer-Orientierung empfangen kann (siehe Kapitel 8).

6.6 Clientprogramm in MATLAB

Die Implementierung des Clientprogramms in MATLAB erfolgte in zwei unterschiedlichen Varianten. Für erste vom Notebook über WLAN gesteuerte Testfahrten wurde ein einfaches Kommandozeilen-Interface „SeekurDrive.m" realisiert, das das Senden von Einzelbefehlen an den Roboter ermöglicht und unter anderem im Laufe der praktischen Tests für die

Überprüfung des mathematischen Modells (siehe Abschnitt 7.1) verwendet wurde. Eine Liste der wichtigsten realisierten Einzelbefehle ist in Tabelle 6.2 ersichtlich, wobei die Befehle je nach zukünftigen Anforderungen änder- und erweiterbar sind. Um die Eingabe für den Benutzer möglichst zu verkürzen wurde jeder Befehl durch einen Buchstaben und die zugehörige Stellgröße umgesetzt.

Eingabe in Kommandozeile	Beschreibung der Roboterbewegung
f <Strecke>	Vorwärtsfahren der Strecke in Millimeter mit maximal eingestellter Translationsgeschwindigkeit
b <Strecke>	Rückwärtsfahren der Strecke in Millimeter mit maximal eingestellter Translationsgeschwindigkeit
l <Winkel>	Drehen nach links um eigene Achse um den Winkel in Grad mit maximal eingestellter Rotationsgeschwindigkeit
r <Winkel>	Drehen nach rechts um eigene Achse um den Winkel in Grad mit maximal eingestellter Rotationsgeschwindigkeit
y <Rotationsgeschwindigkeit>	Fahren einer Kurve nach links mit eingestellter Translationsgeschwindigkeit und Drehung um den Winkel in Grad pro Sekunde
x <Rotationsgeschwindigkeit>	Fahren einer Kurve nach rechts mit eingestellter Translationsgeschwindigkeit und Drehung um den Winkel in Grad pro Sekunde
d <Translationsgeschwindigkeit>	Fahren mit gleichmäßiger Translationsgeschwindigkeit
c	Alle Bewegungen des Roboters stoppen (Translationsgeschwindigkeit, Rotationsgeschwindigkeit = 0)
a <Winkel>	Setzen der Rotationsgeschwindigkeit (Drehrichtung links)
s <Winkel>	Setzen der Rotationsgeschwindigkeit (Drehrichtung rechts)

Tab. 6.2: Einzelbefehlsliste des Kommandozeileninterfaces

Die zweite Variante des MATLAB-Clientprogramms „SeekurTrack.m" dient mithilfe des Trackings der Totalstation als Regelungsprogramm zum Abfahren vordefinierter Trajektorien und somit der praktischen Erprobung in der Arbeit.

Zu Beginn erfolgt die Definition der Soll-Trajektorie laut Abschnitt 6.2 mit den Elementen Gerade und Kreisbogen. Nach dem Verbindungsaufbau über die GeoCOM-Schnittstelle zur

Totalstation und dem Setzen der notwendigen Tachymetereinstellungen (siehe Abschnitt 6.3) wird die Trackingfunktion gestartet, die Socketverbindung zum Serverprogramm aufgebaut und der Roboter in Bewegung gesetzt.

Auf die Algorithmen und Kriterien zur Roboterbewegung entlang der Soll-Trajektorie wird in Kapitel 5 näher eingegangen.

7 Testfahrten

7.1 Überprüfung des mathematischen Modells

7.1.1 Testszenario

Um das hergeleitete mathematische Modell aus Kapitel 4 zu überprüfen, wurden wie in Abbildung 7.1 zu sehen, zwei Leica Standard-Rundprismen auf der Rahmenhalterung über der Längsachse des Roboters befestigt - eines über dem Mittelpunkt, das zweite mit dem Abstand 0,49 m am vorderen Ende der Rahmenhalterung. Während der Messungen bewegte sich der Roboter im *Stop and Go-Modus*, die Steuerung erfolgte über das MATLAB-Interface vom Laptop über WLAN. Für den Versuchsaufbau wurden außerdem die Totalstation *Leica TCRP1201+* im lokalen Koordinatensystem (siehe Abschnitt 6.1) frei stationiert und jeweils mittels manueller Anzielung und IR-Standardmessung die Koordinaten der beiden Prismen am Display abgelesen. Daraus wurde nach jeder einzeln ausgeführten Befehlssequenz die Orientierung des ruhenden Roboters im lokalen Koordinatensystem bestimmt. Die Befehlssequenz definierte eine kreisförmige Bewegung für jeweils ein bestimmtes Zeitintervall (etwa eine Sekunde), wobei die Translationsgeschwindigkeit von 100 mm/sec in drei verschiedenen Testfahrten mit den Rotationsgeschwindigkeiten 5, 10 und 15 °/sec kombiniert wurde.

Da die bei der Ausführung der gleichen Befehlssequenz am Roboter benötigte Zeitdauer nicht konstant ist (siehe Kapitel 2) wurde zur besseren Vergleichbarkeit für alle weiteren Modellberechnungen ein einheitlicher Zeitwert von $1,122$ Sekunden verwendet. Dieser Wert stammt aus gemittelten Zeitmessungen beim Ausführen von genau einer Befehlssequenz. Durch diese Zeitdauer-Unterschiede ergibt sich zum Beispiel bei der Testfahrt (20 Einzelfahrten im *Stop and Go-Modus* mit einer Translationsgeschwindigkeit von 100 mm/sec und einer Rotationsgeschwindigkeit von 5°/sec) ein mittlerer Abstand der zwi-

schen der Einzelfahrten gemessenen Prismenkoordinaten von 99,9 mm mit der einfachen Standardabweichung von 7,2 mm.

Abb. 7.1: Roboter mit zwei Leica Standard-Rundprismen zur Überprüfung des mathematischen Modells

7.1.2 Vergleich des mathematischen Modells mit Roboterfahrt

Der Vergleich des mathematischen Modells mit den Tachymetermessungen erlaubt im Postprocessing mehrere Kombinationsmöglichkeiten zur Berechnung der jeweils nächsten Prismenkoordinaten und der Orientierung. So können zum Beispiel in jedem Berechnungsschritt des Modells die Ergebnisse des vorherigen Schrittes als Anfangskoordinaten und -orientierung übernommen oder stattdessen Updates der Tachymetermessungen verwendet werden. Die Berechnung der Orientierung aus Tachymeterdaten ist natürlich nur bei Testmessungen mit zwei Prismen möglich. Im Regelungsbetrieb muss diese aus den letzten gemessenen Prismenkoordinaten abgeleitet oder den Odometermessungen des Roboters entnommen werden. Die Umrechnung der vom Odometer gemessenen Weglängen der Räder in Orientierungsänderungen erfolgt durch den internen Rechner. Beide Methoden wurden im Rahmen der Diplomarbeit aber nicht näher hinsichtlich deren Genauigkeit untersucht.

Die in den folgenden Unterabschnitten gezeigten Kombinationsmöglichkeiten werden im Postprocessing wie folgt ausgeführt:

Im Fall 1 wird das mathematische Modell mit den Prismenpositionen der Roboterfahrt verglichen. Die Startwerte (Koordinaten und Orientierung des Roboters) für jeden Berechnungsschritt werden jeweils direkt vom letzten Berechnungsschritt übernommen und mit den tatsächlichen Roboterbewegungen verglichen.

Beim Fall 2 (Update der Anfangsorientierung) erhält das mathematische Modell bei jedem Berechnungsschritt die berechnete Orientierung des Roboters aus den beiden Prismenkoordinaten als Startorientierung. Die Startkoordinaten werden jeweils aus der Berechnung des mathematischen Modells übernommen.

Der Fall 3 (Update der Anfangskoordinaten) ist sozusagen die Umkehrung des 2. Falls: Für jeden Berechnungsschritt werden die Startkoordinaten mit den tachymetrisch bekannten Prismenkoordinaten gestützt, die Startorientierung wird aus dem letzten Berechnungsschritt des mathematischen Modells übernommen.

Schließlich werden im Fall 4 (Update der Anfangskoordinaten und der Anfangsorientierung) sowohl die Startkoordinaten als auch die Startorientierung eines Berechnungsschrittes aus den Tachymetermessungen verwendet. Die Berechnung der nächstfolgenden Prismenposition und der neuen Orientierung des Roboters erfolgt durch das mathematische Modell, welche anschließend mit den tatsächlichen Roboterbewegungen verglichen werden können.

In den nachfolgenden Abbildungen wird die Kurvenfahrt des Roboters mit der Translationsgeschwindigkeit von 100 mm/sec und gleichzeitiger Rotation mit 5 °/sec durch die beiden Prismenpositionen „Mitte" und „Vorne" dargestellt und jeweils mit einer der angeführten Berechnungsmöglichkeiten gegenübergestellt.

7.1.2.1 Fall 1: Mathematisches Modell

Im Fall des mathematischen Modells gehen die Ergebnisse einer Berechnung (Koordinaten der Prismenposition und Orientierung des Roboters) als Startparameter in den nächsten Berechnungsschritt ein. Der Vergleich zu den gemessenen Koordinaten der Trajektorie ist in Abbildung 7.2 dargestellt. Wie auch in den nachfolgenden Fällen 2 bis 4 erfolgt der Start der Modellberechnung erst beim zweiten gemessenen Prismenpunkt, da die Orientierungsänderung bei der ersten ausgeführten Roboterbewegung deutlich kleiner ist als bei den nachfolgenden. Der Grund dafür wird in einem Leerlauf der Antriebsriemen vermutet, wenn sich der Roboter das erste Mal in Bewegung setzt. Dieses Phänomen

tritt bei allen weiteren Bewegungsschritten nicht auf, da sich der Roboter in der Folge im „Fahrmodus" befindet und die Spannung in den beiden Antriebsriemen aufrecht bleibt.

Bei beiden Prismenpositionen kann man erkennen, dass der berechnete Kurvenradius kleiner als der tatsächlich gefahrene Radius des Roboters ist, die theoretische Kurvenbogenlänge jedoch größer. Betrachtet man die Orientierung der jeweils zusammengehörigen Kurven, so erkennt man bei der Prismenposition „Mitte" zwar eine sehr gute Anfangsorientierung. Die zu schnell größer werdende berechnete Richtungsänderung führt allerdings wie bei der Prismenposition „Vorne" zu einer klaren Abweichung zur gefahrenen Kurve, allerdings wird vorne auch eine zu geringe Anfangsorientierung angesetzt.

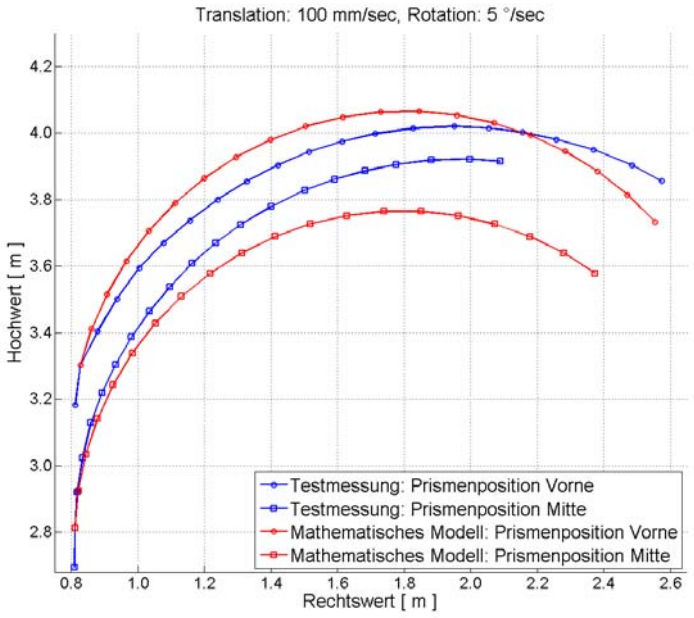


Abb. 7.2: Vergleich des mathematischen Modells zur Testmessung

Die Abweichungen des mathematischen Modells zur tatsächlich gefahrenen Strecke veranschaulicht Abbildung 7.3. Berechnet wird die Abweichung als Abstand zwischen der jeweiligen berechneten und gemessenen Prismenposition. Auf die Angabe der radialen Abweichung wird verzichtet, da die berechneten und gemessenen Kurven einen unterschiedlichen Mittelpunkt haben. Die deutlich größeren Differenzen bei der Prismenposition „Mitte" können aufgrund der Startorientierung beim ersten Berechnungsschritt und den Orientierungsänderungen ähnlich wie im letzten Absatz erklärt werden.

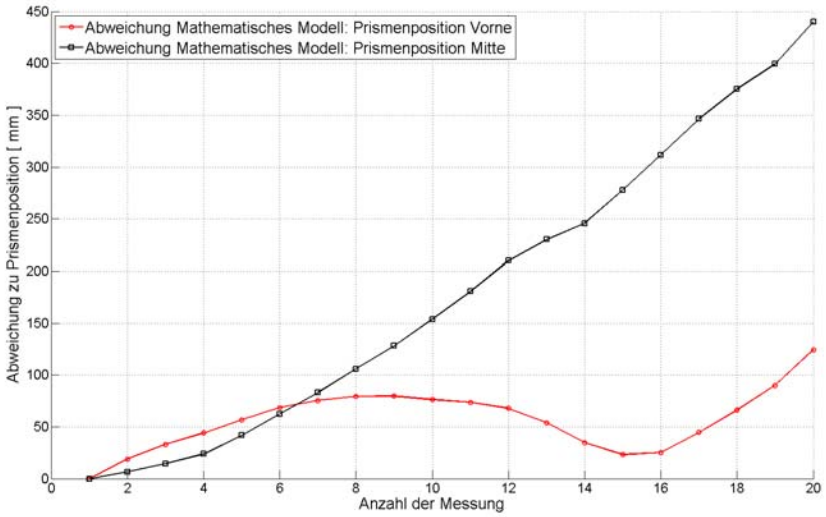


Abb. 7.3: Abweichungen des mathematischen Modells zu den Prismenpositionen der Testmessung

7.1.2.2 Fall 2: Update der Anfangsorientierung

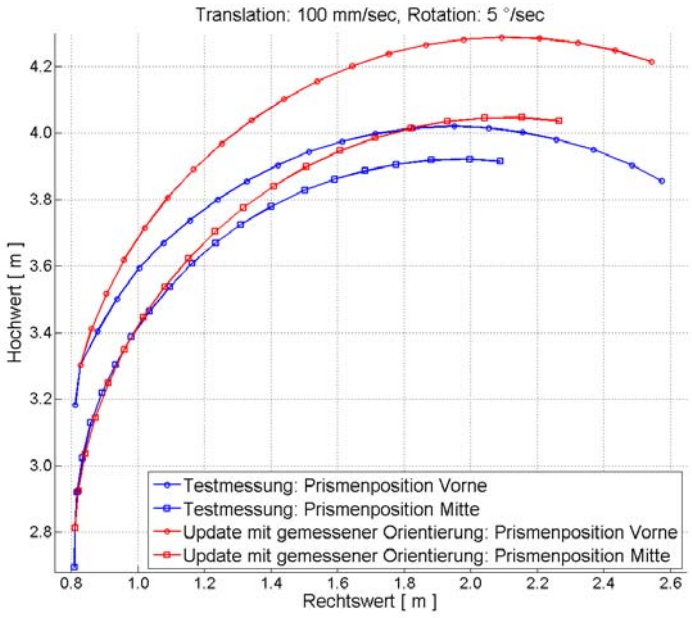


Abb. 7.4: Vergleich des mathematischen Modells (mit Update der gemessenen Orientierung) zur Testmessung

Beim zweiten Fall des Berechnungsmodells erfolgt ein Update der gemessenen Orientierung, indem die aus den beiden Prismenkoordinaten bekannte Orientierung als Startwert in den nächsten Berechnungsschritt eingeht. Die berechneten Koordinaten des Prismas werden allerdings von Schritt zu Schritt aus dem mathematischen Modell übernommen. Dieses Update liefert wie in Abbildung 7.4 zu erkennen ist vor allem für die Prismenposition „Vorne" eine Verschlechterung des Modells.

Die Abweichungen dieses Berechnungsmodells zu den gemessenen Koordinaten zeigt Abbildung 7.5 in der gleichen Skalierung wie Abbildung 7.3.

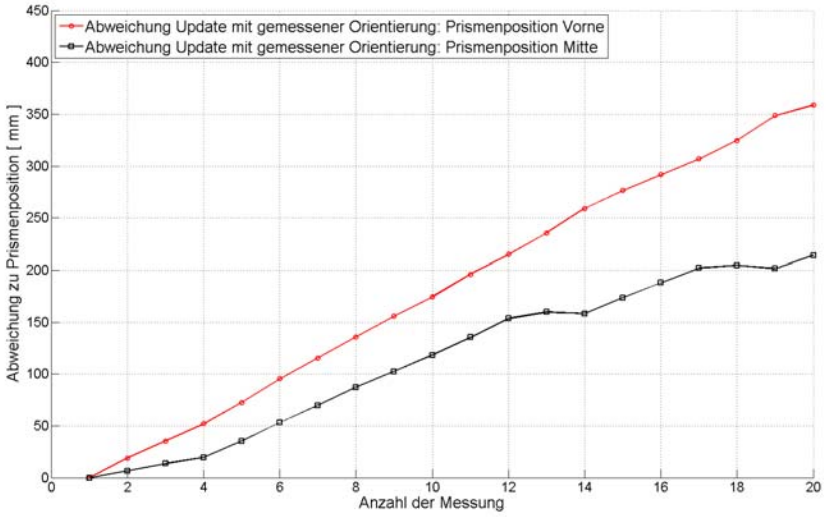


Abb. 7.5: Abweichungen des mathematischen Modells (mit Update der gemessenen Orientierung) zu den Prismenpositionen der Testmessung

7.1.2.3 Fall 3: Update der Anfangskoordinaten

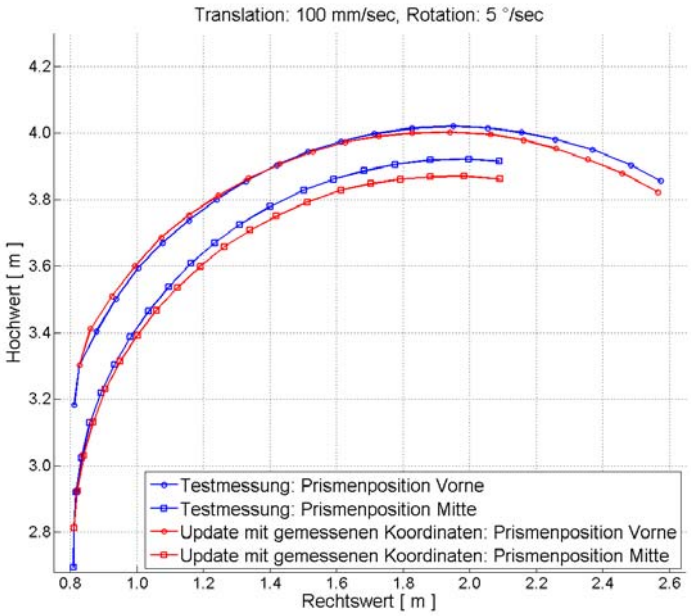


Abb. 7.6: Vergleich des mathematischen Modells (mit Update der gemessenen Koordinaten) zur Testmessung

Das in Abbildung 7.6 zu Grunde liegende Modell verwendet die gemessenen Prismenkoordinaten als Startwert für den jeweils nächsten Berechnungsschritt, die Orientierung wird aus dem mathematischen Modell abgeleitet. Diese Variante liefert daher eine viel bessere Annäherung des Modells an die gefahrene Kurve, welche durch die Abweichungen in Abbildung 7.7 (andere Skalierung als bei den ersten beiden Fällen!) verdeutlicht wird.

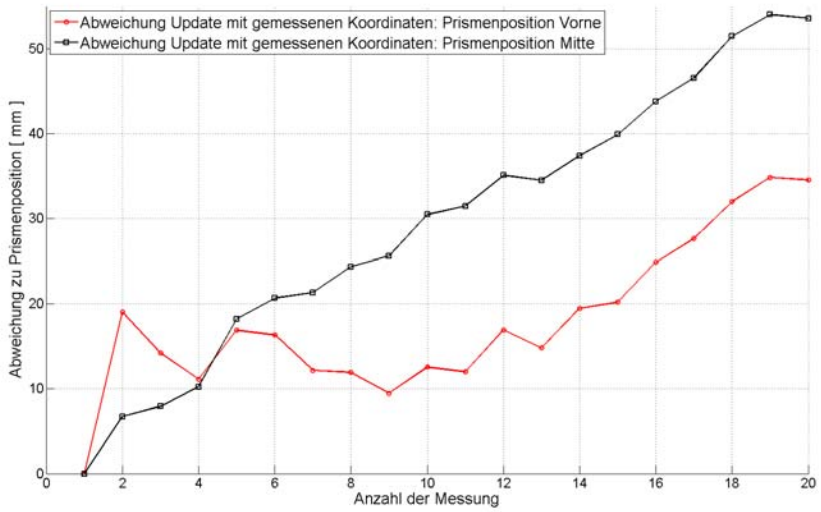


Abb. 7.7: Abweichungen des mathematischen Modells (mit Update der gemessenen Koordinaten) zu den Prismenpositionen der Testmessung

7.1.2.4 Fall 4: Update der Anfangskoordinaten und -orientierung

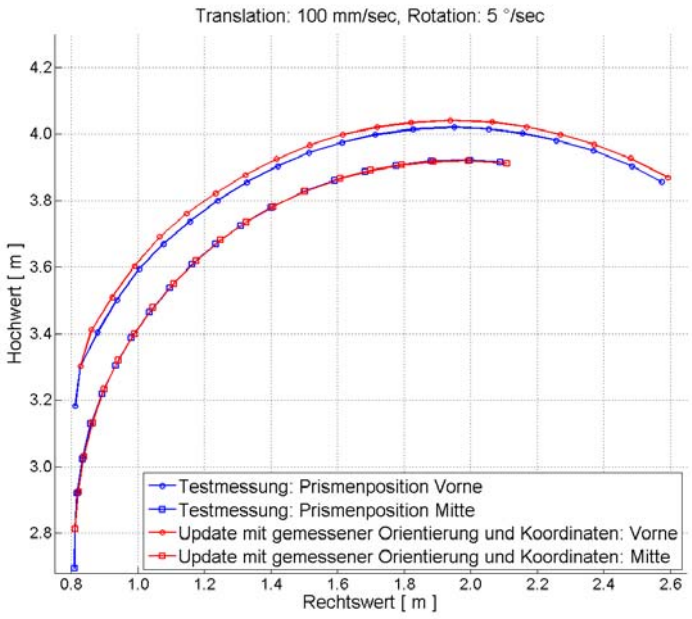


Abb. 7.8: Vergleich des mathematischen Modells (mit Update der gemessenen Orientierung und Koordinaten) zur Testmessung

Die Kombination der beiden letzten Varianten liefert, wie erwartet, das bisher beste Ergebnis. Dabei gehen jeweils die gemessenen Prismenkoordinaten und die daraus abgeleitete Orientierung des Roboters als Startwerte in den nächsten Berechnungsschritt ein. Anders als Abbildung 7.8 vermuten lässt weist die Prismenpositionen „Mitte" ähnlich große Abweichungen auf wie „Vorne" (siehe Abbildung 7.9). Dies resultiert einerseits aus der Berechnungsart der Abweichungen, andererseits kommt die eingangs erwähnte unterschiedliche Ausführungsdauer der Befehlssequenz zur Geltung.

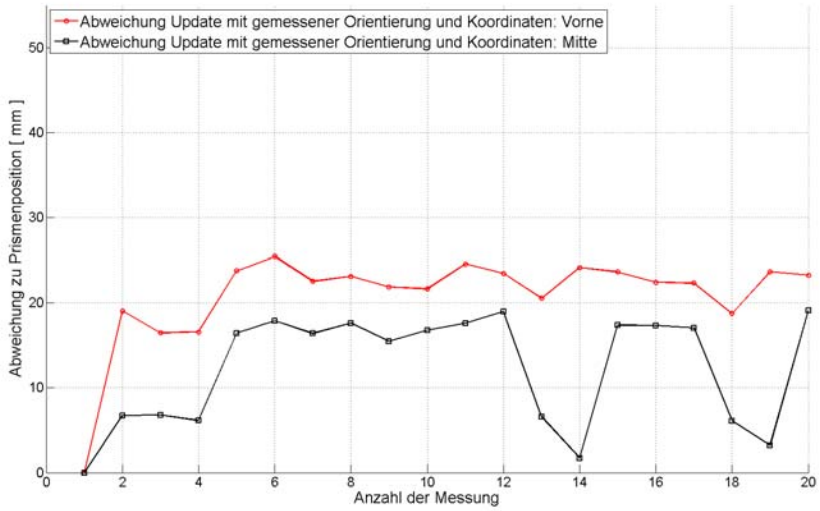


Abb. 7.9: Abweichungen des mathematischen Modells (mit Update der gemessenen Orientierung und Koordinaten) zu den Prismenpositionen der Testmessung

7.1.3 Korrigiertes mathematisches Modell

Das ursprüngliche mathematische Modell in Abbildung 7.2 repräsentiert trotz rein geometrischer Herleitung eine recht gute erste Näherung der tatsächlichen Roboterbewegung, kann aber durch Kalibrierung der Kurvenradien (siehe Abschnitt 7.3) weiter verbessert werden. Betrachtet man den Zusammenhang zwischen R_{Kv} und ω_D in Gleichung 4.18 auf Seite 24, so kann vom kalibrierten Kurvenradius eine korrigierte Drehgeschwindigkeit abgeleitet werden. Dieser Schritt ist notwendig, da ω_D als Eingangsgröße in das mathematische Modell eingeht und neben dem mittleren gefahrenen Kurvenradius R_{Kv} auch alle weiteren Parameter daraus berechnet werden.

Außerdem wird dadurch bestätigt, dass eine rein geometrische Umrechnung der Roboter-

Rotationsgeschwindigkeit ω_D in die Rotationsgeschwindigkeit ω_R der Räder (wie in Abschnitt 4.1) nicht der Realität entspricht. Mit einem einfachen Test kann diese Abweichung auch am Roboter gezeigt werden: Dazu wird der Roboter am Stand um einen bestimmten Winkel (zum Beispiel 90 °) gedreht und der Drehwinkel der Räder ungefähr mit 220 ° durch die Beobachtung der Radventilstellung abgeschätzt. Laut mathematischem Modell wäre für eine Roboter-Drehung um 90 ° allerdings nur eine Drehung der Räder um 189,4 ° notwendig. Erklärt werden kann dieser Unterschied durch die fehlende Lenkbarkeit der Räder, wodurch eine Räderbewegung in Tangentialrichtung zum jeweiligen Kreispunkt nicht möglich ist und die Räder stärker rotieren müssen, um die geforderte Gesamtdrehung zu erreichen.

In Abbildung 7.10 wird das korrigierte mathematische Modell mit den Prismenmessungen verglichen. Neben der Anpassung von ω_D und R_K ist bei der Prismenposition „Vorne" außerdem eine Korrektur der Startorientierung von $11,46°$ notwendig. Dieser Wert wurde nachträglich durch Angleichen der korrigierten mathematischen Modellkurve an die gemessenen Prismenkoordinaten empirisch festgelegt.

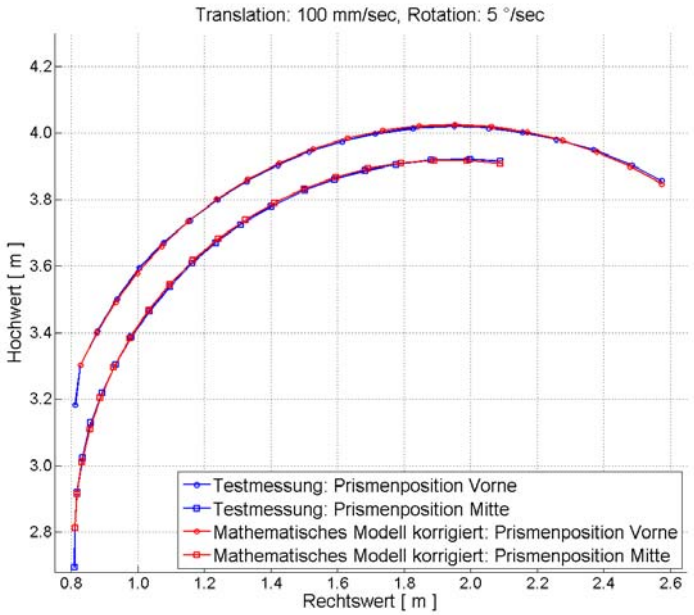


Abb. 7.10: Vergleich des korrigierten mathematischen Modells zur Testmessung

Trotz sehr geringer radialer Abweichungen zwischen dem korrigierten mathematischen Modell und der Kurvenfahrt des Roboters ergeben sich in Abbildung 7.11 noch folgende

Abweichungen zwischen den berechneten und den gemessenen Prismenkoordinaten. Diese können wiederum durch die unterschiedliche Zeitdauer der Bewegungsausführung begründet werden.

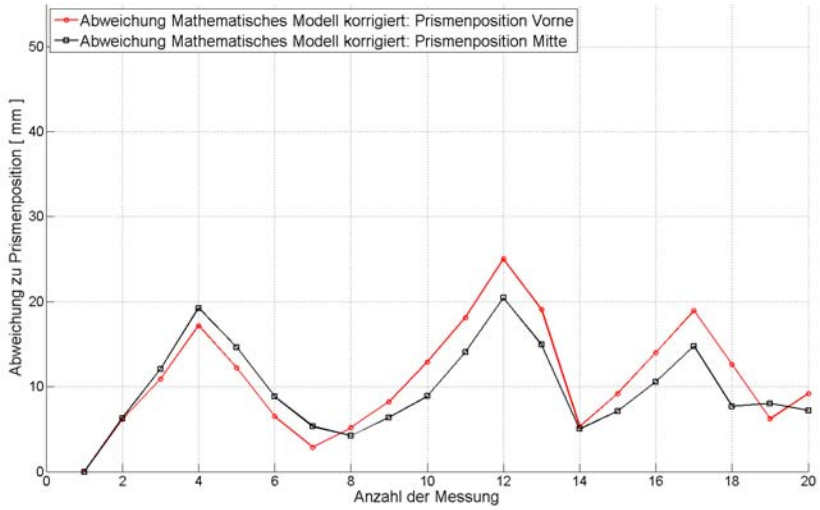


Abb. 7.11: Abweichungen des korrigierten mathematischen Modells zu den Prismenpositionen der Testmessung

7.2 Test der Trackingsoftware

Die entwickelte MATLAB-Routine zur Darstellung und Speicherung der Tachymetertrackingdaten soll anhand eines einfachen Tests geprüft werden. Dazu wurde der Roboter entlang einer kreisförmigen Strecke bewegt, die Koordinaten des auf der Rahmenhalterung befestigten 360°-Prismas wurden von der Totalstation aufgezeichnet und in der Trackingsoftware wie in Abbildung 7.12 dargestellt. Da die Fahrtstrecke selbst nicht exakt einer geschlossenen Kreisbahn folgt (wie in der Abbildung 7.12 oben zu erkennen ist), wird auf einen Vergleich der radialen Abweichungen der Messpunkte zum mittleren gefahrenen Radius verzichtet. Vielmehr soll durch die Testmessung eine Plausibilitätsprüfung der Trackingsoftware hinsichtlich grober Messfehler erfolgen, welche bei der Entwicklung durch Verwendung ungeeigneter GeoCOM-Befehle beobachtet werden konnten.

Für genauere Untersuchungen hinsichtlich der Genauigkeit von zielverfolgenden Tachymetern (speziell auch bei der Messung von kreisförmigen Objektbewegungen) sei auf die Arbeiten von Stempfhuber und Maurer (2001) verwiesen.

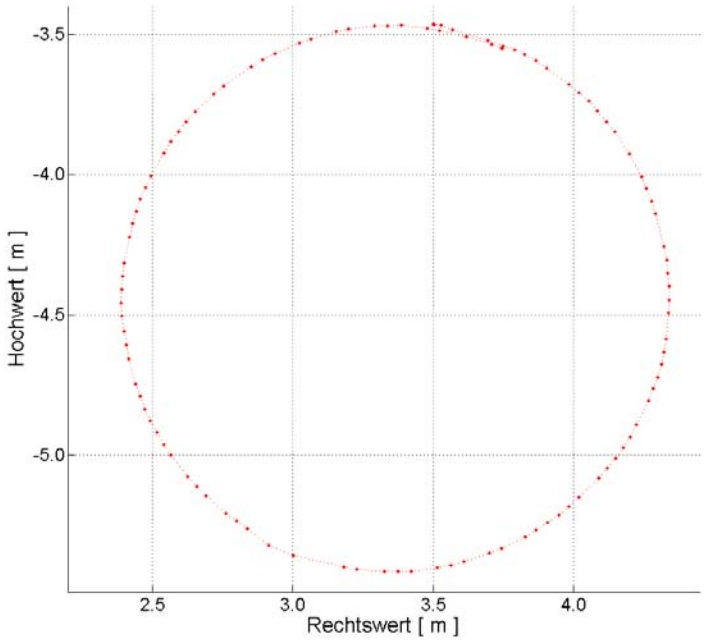


Abb. 7.12: Aufzeichnung der Prismenkoordinaten bei einer Roboter-Kreisfahrt als Trackingsoftwaretest

Um die Fehlereinflüsse durch die Verwendung des 360°-Prismas bei der Robotersteuerung einschätzen zu können erfolgte eine weitere Testmessung. Das Prisma wurde auf der Halterung des ruhenden Roboters montiert, per Hand ein Mal um die vertikale Achse gedreht und dabei die 2D-Koordinaten mit der Tracking-Software in MATLAB aufgezeichnet. Der Abstand zwischen Tachymeter und Prisma wurde mit etwa fünf Metern so gewählt wie der mittlere Abstand später im Anwendungsfall bei der Steuerung des Roboters entlang der Soll-Trajektorie. Die Ergebnisse sind in Abbildung 7.13 dargestellt, wobei sich bei Herausnahme der Ausreißer maximale Abweichungen der Prismenkoordinaten in Richtung des Rechtswertes von 2,6 mm, in Richtung des Hochwertes von 2,8 mm und ein Helmert'scher Punktlagefehler von 1,0 mm ergeben. Recht deutlich erkennt man auch die annähernd quadratische Anordnung der aufgezeichneten Koordinaten, welche aus den periodischen Längs- und Querabweichungen (siehe Abschnitt 6.3) resultiert.

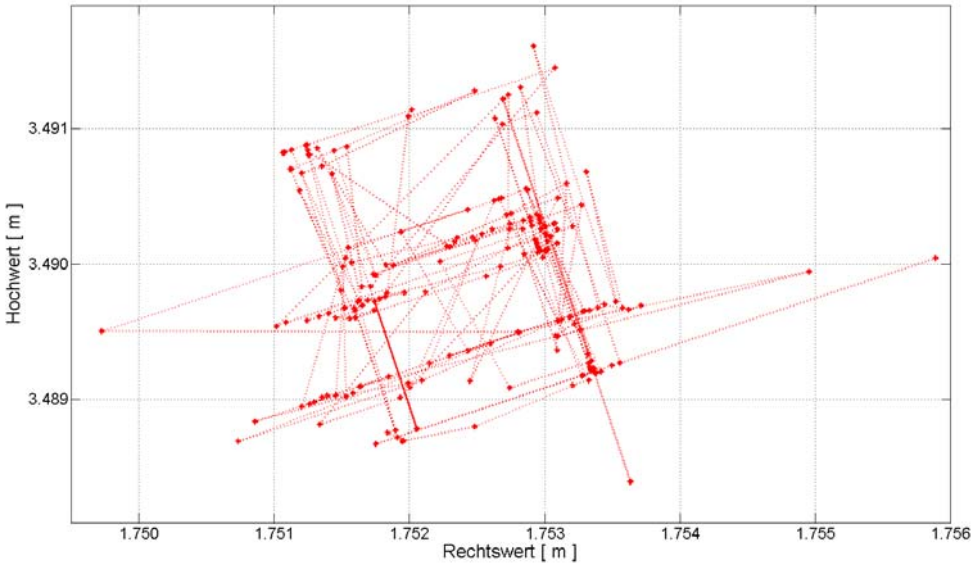


Abb. 7.13: Anordnung der 2D-Koordinaten des 360°-Prismas während der Drehung um die eigene Vertikalachse

7.3 Kalibrierung Kurvenradius

Wie in den Ergebnissen der Überprüfung des mathematischen Modells in Abschnitt 7.1 gezeigt werden kann, liefert die Kalibrierung der Kurvenradien einen maßgeblichen Verbesserungsanteil der Modellbewegung im Vergleich zur gemessenen Kurvenfahrt. Nach dem Starten der Tracking-Software wird der Roboter mit einer gewissen Translations- und Rotationsgeschwindigkeit in Bewegung gesetzt und die 2D-Koordinaten der Prismenpositionen werden als Vektor in MATLAB gespeichert. Mit den Koordinaten kann mittels dem *Gauß-Newton-Verfahren* ein ausgleichender Kreis [1] und somit der jeweilige Kurvenradius berechnet werden. Das grafische Ergebnis einer Kreisausgleichung am Beispiel einer Kurvenfahrt mit 50 mm/sec und 1 °/sec ist in Abbildung 7.14 dargestellt, wobei die roten Punkte die gemessenen Prismenkoordinaten und die blauen Kreise die näherungsweisen und ausgleichenden Kreise darstellen.

[1] http://www.inf.ethz.ch/personal/arbenz/MatlabKurs/node79.html, letzter Zugriff: 17.01.2012

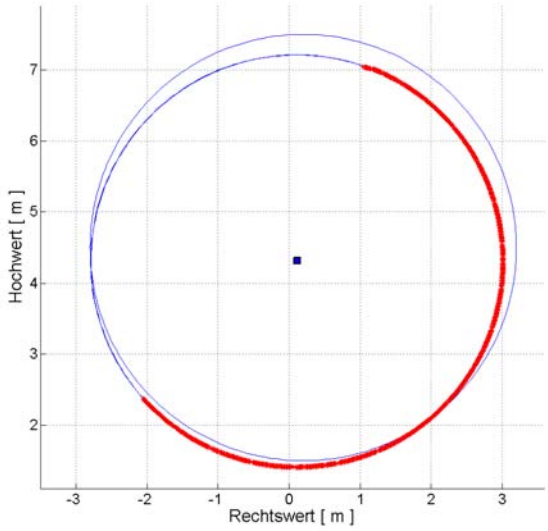


Abb. 7.14: Iteration des ausgleichendes Kreises nach dem *Gauß-Newton-Verfahren* und der Methode der kleinsten Quadrate am Beispiel der Kurvenfahrt mit 50 mm/sec und 1 °/sec.

Die Translationsgeschwindigkeiten 50 mm/sec und 100 mm/sec wurden jeweils mit den Rotationsgeschwindigkeiten 1-7 °/sec in 1°-Intervallen kombiniert, die lineare Geschwindigkeit von 150 mm/sec mit 1-10 °/sec. Höhere Drehgeschwindigkeiten würden den Roboter zum teilweisen Blockieren der Kurveninnenräder zwingen und wurden daher nicht weiter untersucht. Durch die unnatürliche Bewegung bei verhältnismäßig großen Rotations- zu langsamen Translationsgeschwindigkeiten ist es daher auch sinnvoll, Ober- bzw. Untergrenzen und somit kleinstmögliche Kurvenradien zu definieren.

Jede oben angeführte Kombination der Rotations- und Translationsgeschwindigkeiten wurde auch für jeweils drei verschiedene Prismenpositionen auf der Rahmenkonstruktion wiederholt: in der Mitte über dem Drehpunkt des Roboters, über der Vorderachse und vorne an der Spitze. Abbildung 7.15 zeigt die Kreisradien für die Translationsgeschwindigkeit von 50 mm/sec und die drei Prismenpositionen.

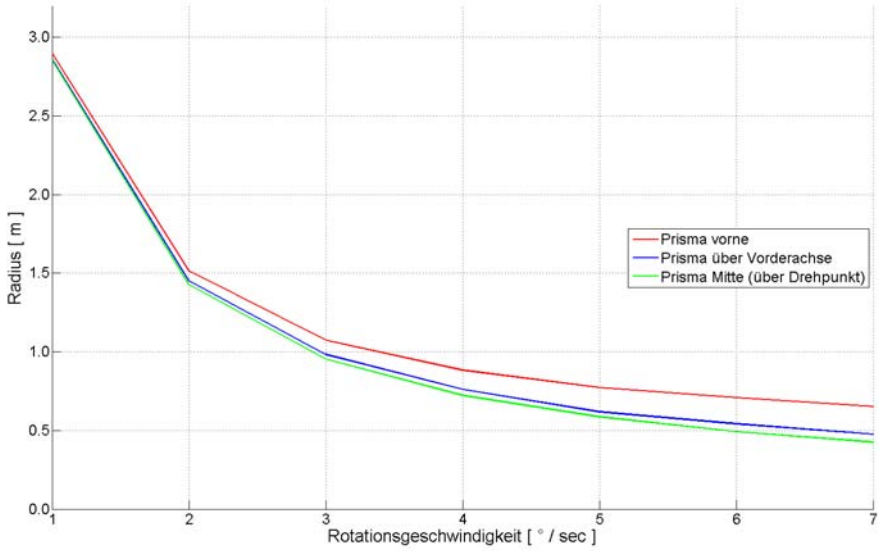


Abb. 7.15: Kreisradiuskalibrierung für die Translationsgeschwindigkeit 50 mm/sec und Rotationsgeschwindigkeiten von 1-7 °/sec

Ein Nachteil für die Steuerung entlang von Kreisbögen ergibt sich aus der Einschränkung, dass nur ganzzahlige Rotationsgeschwindigkeiten eingestellt werden können. Ein Kreisbogen lässt sich also mit gleichmäßiger Geschwindigkeit nur dann ohne Abweichungen abfahren, wenn sein Radius mit einer ganzzahligen Rotationsgeschwindigkeit in der Kalibrierungskurve zusammenfällt. Betrachtet man Abbildung 7.15, so kann zum Beispiel ein Radius von 2,3 m nur durch ständiges Umschalten der Rotationsgeschwindigkeiten von 1 und 2° (bei konstanter Translationsgeschwindigkeit) eingehalten werden.

7.4 Regelverhalten und charakteristische Abweichungen bedingt durch Befestigungsposition des Prismas am Roboter

Durch verschiedene Befestigungspositionen des Rundumprismas am Roboter (in der Mitte über dem Drehpunkt, über der Vorderachse, vorne) ergeben sich dementsprechende Regelverhalten und charakteristische Abweichungen bei Trajektorienelementen und deren Übergängen. In Abbildung 7.16 sind die drei untersuchten Prismenbefestigungspositionen dargestellt, wobei der Abstand zwischen den Prismenzentren über dem Drehpunkt des Roboters und der Vorderachse 21 cm und vom Drehpunkt zum vorderen Ende der Führungsschiene 49 cm beträgt.

Abb. 7.16: Seekur Jr mit dem an drei Stellen befestigten Rundumprisma: Mitte, über der Vorderachse und vorne (Fotomontage)

Anhand von zwei Trajektorien sollen diese Merkmale aufgezeigt und miteinander verglichen werden. Als gleichmäßige Translationsgeschwindigkeit wurde jeweils 50 mm/sec und der idente Regelkreis gewählt, wobei sich die Stellgröße der Regeleinrichtung aus dem Faktor der Geschwindigkeit (50 mm/sec) mit der Regelabweichung in Metern ergibt. In den folgenden Abbildungen ist die vorgegebene Soll-Trajektorie mit blauer Farbe gekennzeichnet und die Anfangs- bzw. Endpunkte mit gelben Quadraten mit schwarzer Umrandung. Bewegt sich der Roboter im definierten Toleranzbereich (± 1 cm) entlang dieser Soll-Linie, werden

die vom Tachymeter aufgenommenen Koordinatenpunkte grün dargestellt, Abweichungen größer als 1 cm in rot.

7.4.1 Trajektorie 1: Zwei Geraden

Die erste Sollstrecke ist knapp über sieben Meter lang und besteht aus zwei Geraden. Der Knickpunkt zwischen den beiden Geraden wurde bewusst gewählt, um das Regelverhalten bei einer abrupten Richtungsänderung der Streckenführung zu untersuchen. Für die Testfahrten wurde der Roboter manuell in seine Startposition geschoben und die Anfangsorientierung ungefähr der Startgeraden angepasst. In Abbildung 7.17 sind die vom Tachymeter aufgezeichneten Koordinaten mit den drei Prismenbefestigungspositionen nebeneinander dargestellt. Sehr deutlich erkennt man, dass die Fahrt mit dem Prisma vorne am Roboter (Abbildung 7.17 links) am besten mit der Vorgabe übereinstimmt. Entlang der beiden Geraden gibt es kaum Abweichungen und auch die Reaktion auf den Knickpunkt erfolgt rascher als im Vergleich zur mittleren Abbildung, wo das Prisma über der Vorderachse befestigt ist. Das Anbringen des Prismas über dem Drehpunkt in der rechten Grafik zeigt, dass sich anfangs kleine Abweichungen immer mehr aufschaukeln. Der Grund dafür liegt in der langsamen Reaktionszeit des proportionalen Reglers infolge des Abstands zwischen Antriebsräder und Roboterzentrum: Selbst wenn die Mitte des Roboters direkt über der Sollstrecke liegt, kann die Roboterlängsachse dazu verschränkt liegen und der vordere Teil einige Zentimeter vom Kurs abweichen. Wenn schließlich auch die Prismenposition außerhalb des Toleranzbereichs liegt, reagiert der Regelkreis und versucht die weiter ausschweifende Fahrlinie der Vorderräder durch eine höhere Rotationsgeschwindigkeit auszugleichen. Dadurch wird aber auch die Längsachsenverschränkung zur Geraden beim nächsten Durchqueren des Toleranzbandes größer und es entsteht der in der Grafik dargestellte Effekt des Aufschaukelns der Abweichungen.

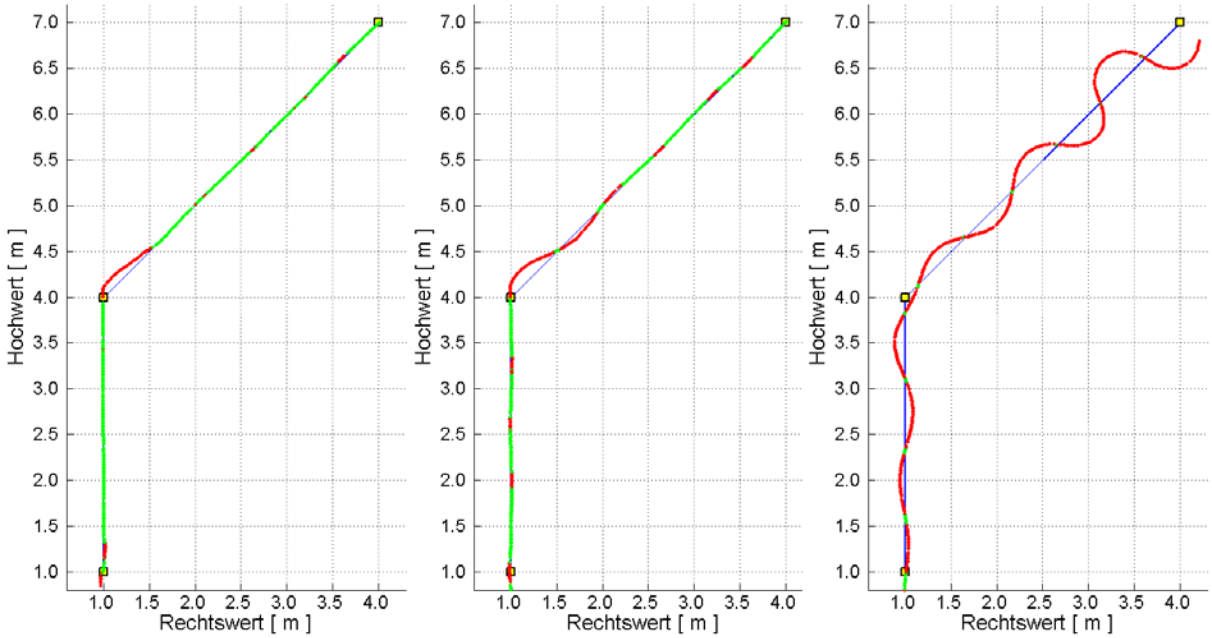


Abb. 7.17: Trajektorie 1: Prismenbefestigung am Roboter vorne, über der Vorderachse, Mitte (Abb. von links nach rechts); blau: Soll-Trajektorie, grün: Roboterposition innerhalb des Toleranzbereichs ±1 cm, rot: außerhalb des Toleranzbereichs

Die nachfolgenden Grafiken (Abbildung 7.18 bis Abbildung 7.20) zeigen die dazugehörigen Abweichungswerte zu den Testfahrten aus Abbildung 7.17. Die in Fahrtrichtung gesehen links neben der Soll-Strecke liegenden Abweichungen sind auf der Hochachse positiv, Abweichungen rechts davon negativ eingezeichnet. Weiters ist die unterschiedliche Achsenskalierung aufgrund der unterschiedlich großen Abweichungen zu beachten.

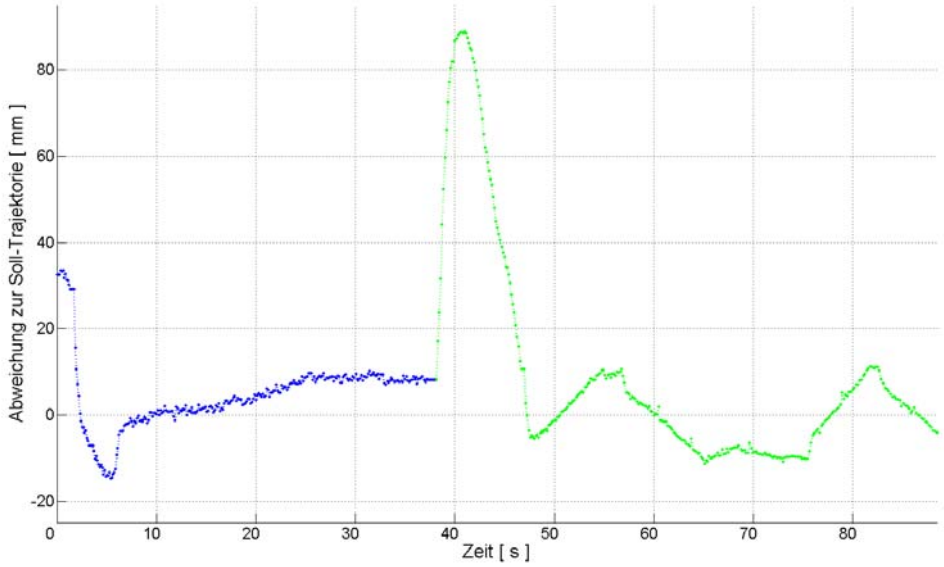


Abb. 7.18: Abweichungen zur Soll-Trajektorie bei Prismenbefestigung vorne am Roboter (blau: 1. Gerade, grün: 2. Gerade)

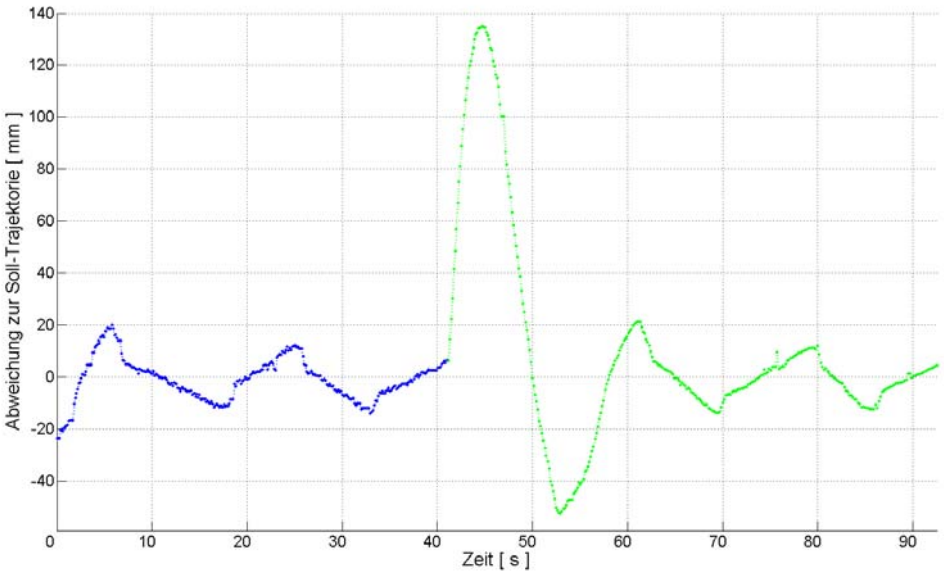


Abb. 7.19: Abweichungen zur Soll-Trajektorie bei Prismenbefestigung über der Vorderachse (blau: 1. Gerade, grün: 2. Gerade)

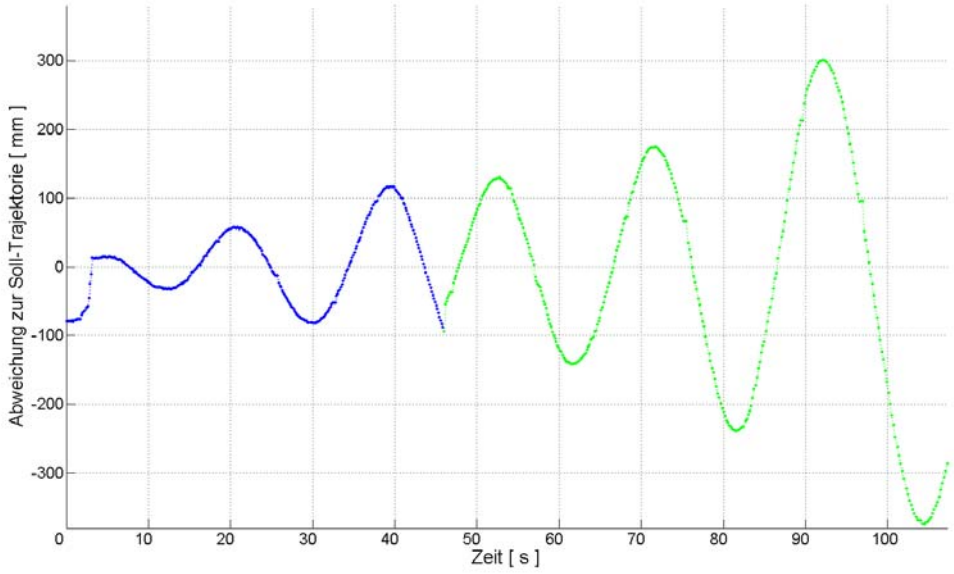


Abb. 7.20: Abweichungen zur Soll-Trajektorie bei Prismenbefestigung über dem Robotermittelpunkt (blau: 1. Gerade, grün: 2. Gerade)

7.4.2 Trajektorie 2: Oval

Zwei Geraden und zwei Halbkreisbögen bilden ein Oval für die zweite Soll-Trajektorie in Abbildung 7.21, wobei jeweils der Punkt links unten Start- und Endpunkt ist und die Fahrt in der Verlängerung der ersten Gerade gestartet wird. In der rechten Grafik bewirkt die Prismenbefestigung über dem Drehpunkt die bereits im vorigen Abschnitt beschriebene Aufschaukelung der Abweichungen und führt sogar soweit, dass der Regelkreis versagt und der Roboter nicht mehr auf die Sollstrecke zurückkehrt. Wiederum steuert der eingesetzte Proportional-Regler den großen Abweichungen durch hohe Rotationsgeschwindigkeiten entgegen, bis der Abstand zur Geraden zu groß wird und die Fahrt abgebrochen werden muss. Beim Vergleich der beiden linken Abbildungen erkennt man leichte Vorteile beim Ausregeln der Geradenabweichungen (Abbildung 7.21 links: Prisma vorne). Bei den Übergängen - sowohl bei Gerade zu Kreisbogen als auch umgekehrt - und bei der Kurvenfahrt selbst weist allerdings die mittlere Grafik mit der Prismenbefestigung über der Vorderachse die kleinsten Differenzen auf.

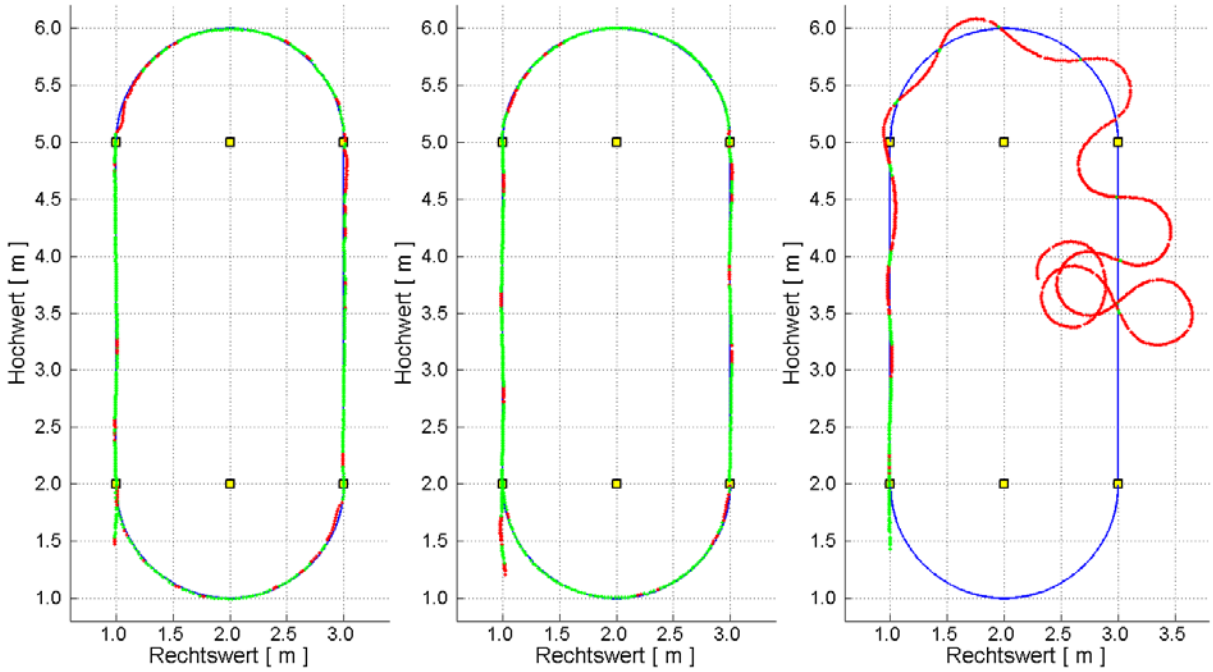


Abb. 7.21: Trajektorie 2: Prismenbefestigung am Roboter vorne, über der Vorderachse, Mitte (Abb. von links nach rechts); blau: Soll-Trajektorie, grün: Roboterposition innerhalb des Toleranzbereichs ±1 cm, rot: außerhalb des Toleranzbereichs

Die Detailansicht Abbildung 7.22 (aus Abbildung 7.21, linke Grafik) beim Übergang zwischen der ersten Geraden zum Kreisbogen soll zeigen, warum die aufgezeichneten Prismenkoordinaten zu Beginn innerhalb des Bogens liegen. Durch den großen Abstand zwischen Drehpunkt des Roboters und Prisma an der Spitze (49 cm) beginnt die Rotation für die Kreisbogenfahrt etwas zu früh. Hat nämlich das Prisma den Endpunkt der Geraden erreicht, beendet der Regelkreis das Trajektorienelement „Gerade" und geht zum nächsten Element über, das in diesem Fall der Kreisbogen ist, und startet die Kurvenfahrt mit der aus der Kurvenradius-Kalibrierung (Abschnitt 7.3) bekannten Rotationsgeschwindigkeit.

Ist das Prisma über der Vorderachse montiert, beträgt der Abstand vom Drehpunkt zum Prismenzentrum nur mehr 21 cm und die Rotation für den Kreisbogen beginnt daher erst später. Außerdem wirkt sich in diesem Fall auch noch eine gewisse Zeitverzögerung positiv auf das Regelverhalten auf, bis der Regelkreis auf die etwas verzögert eintreffenden Tachymeterdaten reagieren und die Steuerbefehle an den Roboter weiterleiten kann.

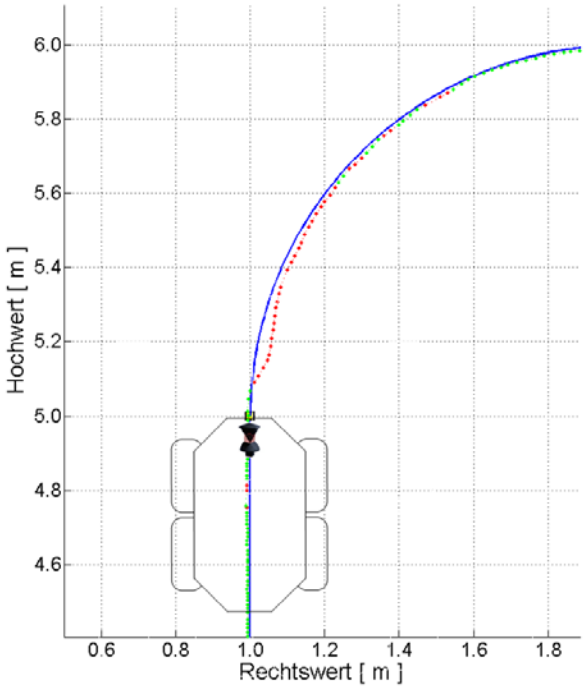


Abb. 7.22: Regelverhalten beim Übergang von Gerade zu Kreisbogen bei Prismenposition „Vorne"; blau: Soll-Trajektorie, grün: Roboterposition innerhalb des Toleranzbereichs ±1 cm, rot: außerhalb des Toleranzbereichs

Auch der Übergang des Kreisbogens zur zweiten Gerade aus Abbildung 7.21 (linke Grafik) ist in Abbildung 7.23 vergrößert dargestellt. Der Roboter durchfährt also von links oben kommend zuerst den Kreisbogen und dann die Gerade. Das Anbringen des Prismas vorne am Roboter bewirkt am Übergangspunkt, dass die Orientierung des Roboters nicht mit der Geradenrichtung übereinstimmt. Der Grund dafür ist, dass der Prismenpunkt den Kreisbogen bereits durchfahren und somit das Abbruchkriterium erfüllt hat, während der Drehpunkt des Roboters noch weiter dahinter liegt. Da die Abweichung am Übergangspunkt und auch am Anfang der Geraden innerhalb der Toleranzgrenze liegt, durchquert der Roboter das Toleranzband mit der entsprechenden Rest-Orientierung am Ende des Kreisbogens und weicht in der Folge in Fahrtrichtung gesehen links von der Gerade ab. Diese Rest-Orientierung ist in Abbildung 7.23 als schwarze Linie angedeutet.

Beim Anbringen des Prismas über der Vorderachse des Roboters wäre das Abbruchkriterium für den Kreisbogen erst später erfüllt und der Roboter hätte beim Übergangspunkt bereits besser die Orientierung der darauf folgenden Gerade erreicht.

Anmerkung: Bei beiden Detailansichten ist der Robotergrundriss nicht maßstabsgetreu

eingezeichnet!

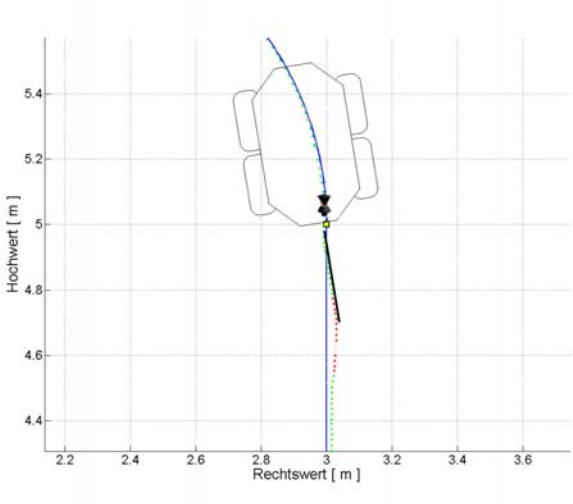


Abb. 7.23: Regelverhalten beim Übergang von Kreisbogen zu Gerade bei Prismenposition „Vorne"; blau: Soll-Trajektorie, grün: Roboterposition innerhalb des Toleranzbereichs ±1 cm, rot: außerhalb des Toleranzbereichs

Für die beiden linken Testfahrten aus Abbildung 7.21 sind die gemessenen Koordinatendifferenzen zum Oval in Abbildung 7.24 und Abbildung 7.25 dargestellt und die vier Trajektorienelemente farblich unterschiedlich gekennzeichnet. Bei der Prismenbefestigung vorne am Roboter ergibt sich ein rms der Abweichungen zur Soll-Trajektorie von 9,3 mm bei einer maximalen Differenz zur Soll-Strecke von 37,1 mm.

Die Testfahrt mit dem Prisma über der Vorderachse bringt eine Verbesserung des rms der Abweichungen auf 7,7 mm und eine Maximalabweichung von 18,7 mm. Wird dabei der Einregelvorgang des Roboters auf die erste Gerade berücksichtigt und werden nur die Messwerte ab dem Startpunkt des Ovals für die Berechnung des rms hergenommen kann dieser auf 6,7 mm reduziert werden.

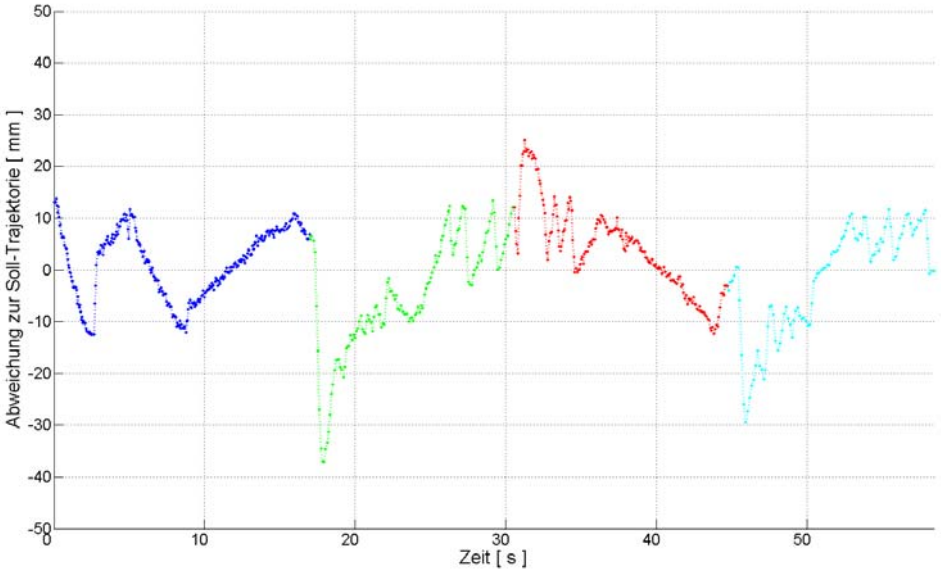


Abb. 7.24: Abweichungen zur Soll-Trajektorie bei Prismenbefestigung vorne am Roboter (blau: 1. Gerade, grün: 1. Kreisbogen, rot: 2. Gerade, cyan: 2. Kreisbogen)

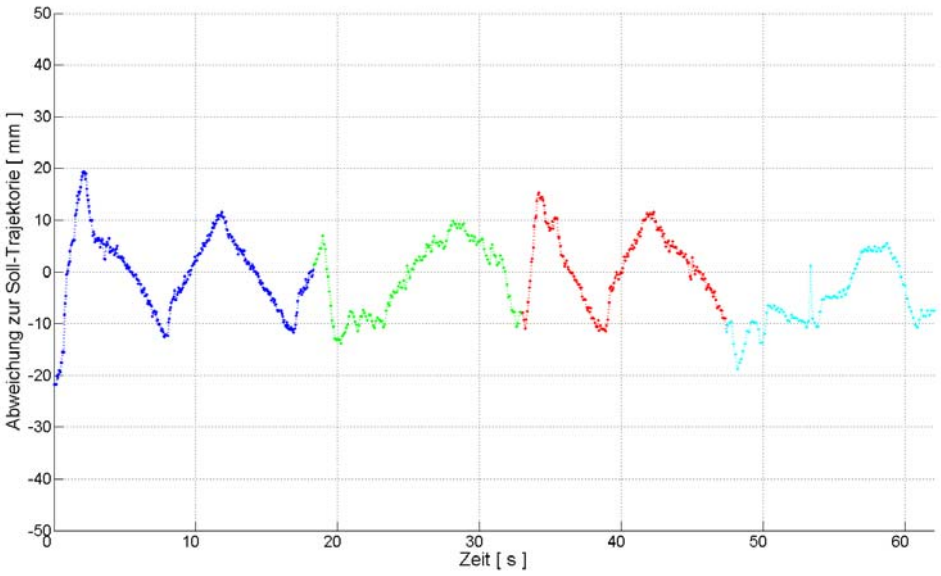


Abb. 7.25: Abweichungen zur Soll-Trajektorie bei Prismenbefestigung über der Vorderachse (blau: 1. Gerade, grün: 1. Kreisbogen, rot: 2. Gerade, cyan: 2. Kreisbogen)

8 Zusammenfassung und Ausblick

Das Ziel der Arbeit war die Entwicklung einer Steuerung für die mobile Roboterplattform der Forschungsgruppe Ingenieurgeodäsie. Der Roboter sollte mithilfe eines Regelkreises eine koordinativ vorgegebene Soll-Trajektorie mit einem rms der Abweichungen in Querrichtung zur Soll-Trajektorie von 1 cm nachfahren können. Die Kommunikation zwischen dem zur Steuerung verwendeten Rechner und dem Roboter sollte über WLAN realisiert werden, wobei der Regelkreis die Trackingdaten eines zielverfolgenden Tachymeters als Eingangsdaten verwenden sollte. Zur Modellierung der fahrdynamischen Eigenschaften der Roboterplattform sollte ein mathematisches Modell entwickelt und getestet werden.

Für die Kommunikation zwischen dem Steuerrechner und dem roboterinternen Rechner beziehungsweise der Steuerung über eine Soll-Trajektorie waren zahlreiche Vorarbeiten notwendig. So wurde am Gang der Forschungsgruppe Ingenieurgeodäsie ein lokales Koordinatensystem definiert und eingemessen und selbstklebende Reflektorfolien als Fixpunkte verwendet. In dem lokalen Koordinatensystem konnten so die Soll-Trajektorien für die Testmessungen koordinativ festgelegt werden. Die TCP/IP-Socketverbindung ermöglichte eine sehr zuverlässliche Möglichkeit, die Steuerdaten vom MATLAB-Steuerungsprogramm an den Roboterrechner zu senden.

Das mathematische Modell lieferte trotz rein geometrischer Herleitung einen guten näherungsweisen Zusammenhang zwischen den verwendeten Steuerbefehlen und den tatsächlichen Positions- und Orientierungsänderungen. Eine deutliche Verbesserung konnte im korrigierten Modell durch die Kalibrierung der Kurvenradien erreicht werden. Als nicht beseitigbares Restproblem blieb die unterschiedliche lange Ausführungsdauer von gleichen Befehlssequenzen. Dies führte auch dazu, dass die ursprünglich geplante Roboteransteuerung mittels Einzelbefehlen nicht für die Navigation entlang der Soll-Trajektorie verwendet werden konnte. Stattdessen wurde am Roboterrechner eine Serverapplikation implementiert, die mittels Multithreading eingehende Steuerbefehle berücksichtigt und so während der Fahrt die Rotationsgeschwindigkeit ändern kann.

Als Befestigungspositionen für das 360°-Prisma wurden drei Varianten untersucht, die in Abhängigkeit der gewählten Geschwindigkeit und Soll-Trajektorie jeweils die kleinsten Regelabweichungen verursachten. Daher kann auch keine Empfehlung hinsichtlich weiterer Testfahrten gegeben werden. Ähnlich wie bei Beetz und Schwieger (2007) könnte das Prisma über dem geometrischen Fahrzeugschwerpunkt montiert und die Koordinaten für die Berechnung der Regelabweichung vorausberechnet werden. Dieser angenommene Berechnungspunkt liegt in der Fahrzeug-Längsachse und kann nur bei bekannter Orientierung des Fahrzeuges bestimmt werden, welche im Fall des Roboters entweder durch die Odometerdaten oder die Schätzung der jeweils zuletzt registrierten Tachymeterkoordinaten bestimmt werden könnte. Die Distanz zwischen dem Fahrzeugschwerpunkt und dem Berechnungspunkt kann beim Befahren einer Gerade größer gewählt werden, wodurch die Regelung eine höhere Systemstabilität erhält. In Kurven darf diese allerdings nicht größer sein als der Fehler, der durch die Tachymeter-Zeitverzögerung hervorgerufen wird.

Ein weiterer Entwicklungspunkt ist die Überprüfung der Abweichungen zur Soll-Strecke mit einem unabhängigen Messverfahren. Derzeit liefert das Tachymeter einerseits die Regelgröße zum Vergleich mit der Führungsgröße und zugleich sind diese berechneten Regelabweichungen auch Basis für die Beurteilung des Regelverhaltens.

Der geforderte rms der Abweichungen in Querrichtung zur Soll-Trajektorie von 1 cm für eine typische Streckenführung eines Ovals wurde im Experiment mit bis zu 6,7 mm erreicht, wobei als Translationsgeschwindigkeit des Roboters 50 mm/sec gewählt wurde. Dieser Abweichungswert konnte bereits mit einem einfachen proportionalen Regler und der gewählten Befestigungsposition des 360°-Prismas über der Vorderachse erzielt werden. Bei den Testfahrten zeigten sich vor allem Schwächen bei der Prismenbefestigung über dem Roboter-Drehpunkt und beim Durchfahren der Übergänge zwischen den Trajektorienelementen. Der entwickelte Regelkreis mit dem proportionalen Regler ist für langsame Geschwindigkeiten bis etwa 150 mm/sec ausreichend, kann aber aufgrund seines Funktionsprinzips keine Restabweichungen ausregeln und erfordert eine lange Ausregelzeit bei abrupten Richtungswechseln der Vorgabestrecke. Zur Minimierung der Regelabweichungen und der schnelleren Reaktion bei Trajektorienelement-Übergängen ist daher bei weiterführenden Arbeiten die Erweiterung zu einem PID-Regler unbedingt notwendig. Die Verbesserung der Regeleinrichtung würde dann auch höhere Geschwindigkeiten ermöglichen. Um die dadurch größer werdenden Kurvenradien durchfahren zu können, müsste allerdings auch ein anderer Versuchsort mit größerer Fläche gewählt werden.

Literaturverzeichnis

Beetz, A., Schwieger, V., 2007. Optimierung von Regelalgorithmen zur Baumaschinensteuerung am Beispiel eines Simulators. Wichmann Verlag, Heidelberg.

Beetz, A., Schwieger, V., 2008. Integration of Controllers for Filter Algorithms for Construction Machine Guidance. First International Conference on Machine Control and Guidance 2008.

Gläser, A., 2007. Modulares System zur Automatisierung hochgenauer geometrischer Positionierung und Bahnführung im Bauwesen. Dissertation, Universität Stuttgart.

Hennes, M., 2000. Zur Bestimmung der Leistungsfähigkeit trackender Totalstationen. Beitrag zum XIII. Kurs für Ingenieurvermessung, München, 13.-17.3.2000.

Hennes, M., Favre, C., 2000. Zum Einfluss der geometrischen Ausrichtung von 360-Grad-Reflektoren bei Messungen mit automatischer Zielerfassung. VPK 2/2000, 72–78.

Ingensand, H., 2001. Systematische Einflüsse auf praktische Messungen mit dem Tachymeter und Digitalnivellier. Qualitätsmanagement in der Geodätischen Messtechnik, DVV Schriftenreihe 42/2001, 120–137.

Leica Geosystems, 2006. Leica TPS1200 GeoCOM Reference Manual v 1.10.

Lunze, J., 2008. Regelungstechnik 1. Springer Verlag.

Merz, L., Jaschek, H., 2003. Grundkurs der Regelungstechnik. Oldenbourg Verlag.

Mitschke, M., Wallentowitz, H., 2004. Dynamik der Kraftfahrzeuge. Springer Verlag, Berlin - Heidelberg.

MobileRobots, 2010. Seekur Jr Operations Manual Version 1.

Rahilly, T., 2006. Einführung in die Multithreading-Programmierung: http://www.zdnet.de/magazin/39197412/einfuehrung-in-die-multithreading-programmierung.htm. ZDNet, Anwendungsentwicklung, letzter Zugriff: 20.01.2012.

Stempfhuber, W., 2004. Ein integritätswahrendes Messsystem für kinematische Anwendungen. Dissertation, TU München.

Stempfhuber, W., Ingensand, H., 2008. Baumaschinenführung und -steuerung - Von der statischen zur kinematischen Absteckung. Zeitschrift für Geodäsie, Geoinformation und Landmanagement 1/2008, 36–44.

Stempfhuber, W., Kirschner, H., 2008. Kinematische Leistungsfähigkeit von zielverfolgenden Tachymetern - ein Beitrag zum Stand der Technik am Beispiel des Leica TPS1200+. Allgemeine Vermessungs-Nachrichten (AVN) 6/2008, 216–223.

Stempfhuber, W., Maurer, W., 2001. Leistungsmerkmale von zielverfolgenden Tachymetern bei dynamischen Applikationen. DVW Schriftenreihe 42/2001, 54. Fortbildungsseminar, Qualitätsmanagement in der Geodätischen Messtechnik, 189–205.

Stempfhuber, W., Schnädelbach, K., Maurer, W., 2000. Genaue Positionierung von bewegten Objekten mit zielverfolgenden Tachymetern. Ingenieurvermessung 2000, XIII. International Course on Engineering Surveying, 144–154.

Tröster, F., 2011. Steuerungs- und Regelungstechnik für Ingenieure. Oldenbourg Wissenschaftsverlag GmbH.

Willemer, A., 2006. Client-Server Socketprogrammierung: http://www.willemer.de/informatik/unix/unprsock.htm, letzter Zugriff: 20.01.2012.

Zimmermann, D., Konrad, B., 2003. Dreidimensional gesteuerte Baumaschinen für den Verkehrswegebau. Vermessung, Photogrammetrie, Kulturtechnik 3/2003, 104–109.