OPTISCH GEFÜHRTES STEUERUNGSKONZEPT EINES ROBOTERS IN ECHTZEIT

Dissertation

zur Erlangung des akademischen Grades

Doktoringenieur
(Dr.-Ing.)

von Dipl.-Ing. Alexander Krasnych

am 28.07.2004 vorgelegte Dissertation

geb. am 07.06.1975 in Leninogorsk

genehmigt durch die Fakultät Elektrotechnik und Informationstechnik

der Otto-von-Guericke-Universität Magdeburg

Gutachter: Prof. Dr.-Ing. habil. Christian Döschner (Otto-von-Guericke Universität Magdeburg)
Prof. Dr.-Ing. Markus Krabbes (FH Leipzig)
Prof. Dr.-Ing. Alexander Chorchordin (Staatliche technische Universität Donezk)

Promotionskolloquium am 21.02.2005

Danksagung

Die vorliegende Arbeit entstand im Rahmen meiner Tätigkeit als Promotionsstudent am Institut für Automatisierungstechnik der Otto-von-Guericke Universität Magdeburg.

Mein besonderer Dank gilt Herrn Prof. Dr.-Ing. habil. Christian Döschner für die Möglichkeit, diese Arbeit an seinem Institut durchführen zu können. Er hat diese Arbeit mit stetem Interesse, wertvollen Hinweisen und freundlicher Unterstützung begleitet und gefördert.

Danken möchte ich auch allen meinen Kollegen am Institut, vor allem Prof. Dr.-Ing. Markus Krabbes für viele anregende Diskussionen und die kritische Durchsicht des Manuskripts, sowie Dipl.-Ing. Bernd Kupfernagel und Dr.-Ing. Günther Müller für die Hilfe bei der Erstellung der technischen Hilfsmittel.

Weiterer Dank sei allen Mitarbeitern des Instituts ausgesprochen, deren Türen mir stets offen standen für fachliche Diskussionen, praktischen Rat und persönliche Ermunterung.

Der größte Dank gebührt meiner Mutter für ihre Geduld, ihr Verständnis und ihre Unterstützung, die sie mir über all die Jahre mit ganzer Kraft entgegengebracht hat.

Zusammenfassung

Zukünftige Anwendungen erfordern eine Weiterentwicklung der Roboterfähigkeiten hinsichtlich der Autonomie. Das Ziel dabei ist die Realisierung intelligenter sehender Roboter, um die verschiedenen industriellen Aufgaben auszuführen. Deshalb wird in dieser Arbeit eine Struktur der optisch geführten Robotersteuerung für die Objektverfolgung entwickelt und untersucht.
Aufgrund der Beschränkung der derzeit industriell gegebenen Bildverarbeitungssysteme wurde ein ereignisdiskretes visuelles Steuerungskonzept vorgestellt und untersucht. Dieses Konzept der Steuerung wurde für zwei am Objekt festpositionierten Kameras zur präzisen Bestimmung der Position und der Orientierung des Endeffektors entwickelt. Für die in der Literatur beschriebenen Verfahren zur visuellen Steuerung sind die Bildverarbeitung, die aufwendigen Kalibrierungen der Kameras und kinematische Transformationen erforderlich und realisiert.

Die Kinematik des Roboters mit deren Nichtlinearität und Mehrdeutigkeit spielt bei der optisch geführten Robotersteuerung eine wesentliche Rolle. Bei Robotern geht es dabei insbesondere um die Beziehung zwischen den Angaben der Winkelkoordinaten der Antriebsachsen und den kartesischen Koordinaten speziell des Endeffektors. Dabei unterscheidet man zwischen der direkten und inversen Kinematik. Die direkte Kinematik ordnet einer bestimmten Stellung seiner Gelenke, d.h. einer Konfiguration, eine Position und Orientierung des Endeffektors im kartesischen Arbeitsraum zu. In dieser Arbeit wurde die Vorgehensweise zur Berechnung der direkten Kinematik erörtert und sowohl simulativ als auch experimentell für den redundanten Roboter *amtec r7* untersucht.
Da die inverse Kinematik ein sehr komplexes Problem wegen seiner starken Nichtlinearität und Mehrdeutigkeit darstellt, wurde sie mittels eines speziell angepassten Levenberg-Marquardt Optimierungsverfahrens berechnet. Ein Vorteil dieses Verfahrens besteht darin, dass es die Ermittlung der Position und Orientierung des Endeffektors *Online* ermöglicht. Die Effektivität dieser entwickelten Methode wurde sowohl simulativ als auch experimentell für den 7-achsigen Roboter nachgewiesen.

Ein weiteres Problem der visuellen Robotersteuerung stellt die Trajektoriengenerierung im kartesischen Koordinatensystem dar. Die Trajektorienformen sind von den jeweils gegebenen Aufgaben abhängig. Entsprechend der entwickelten Struktur der visuellen Robotersteuerung sind linearen Trajektorien im kartesischen Raum notwendig, die gemäß dem Polynom fünfter Ordnung berechnet wurden. Diese Trajektorien wurden jeweils zwischen den Anfangs- und Zielpunkten des Endeffektors im Arbeitsraum des Roboters gebildet.

Um die flexibleren Fertigungsprozesse zu gestalten, ist der Einsatz der optischen Sensoren zur Erfassung der Umgebung des Roboters erforderlich. Zur Bestimmung der Position und Orientierung von Zielobjekten wurden in dieser Arbeit positionsempfindliche Detektoren (PSD) wegen ihrer hohen Genauigkeit, ihrer Schnelligkeit und der kleinen Nichtlinearität eingesetzt. Zur Ermittlung der Position und Orientierung des Objektes mittels visueller Information wurden 4 Laserdioden auf dem Endeffektor befestigt. Um die Laserdioden-Wirkung mit den PSD-Sensoren zu erfassen, wurden die Modulationssignale für die Laserdioden mit Hilfe des Rechners erzeugt. Da in der visuellen Steuerung zur Erfassung des Objektes Kameras eingesetzt wurden, müssen die verwendeten Kameras kalibriert werden, um

Korrespondenz zwischen Punkten auf den zweidimensionalen Bildern und den Punkten der darin dargestellten Objekte in der dreidimensionalen Welt herzustellen. Dabei gilt es, die inneren und äußeren Kameraparameter, die sich aus einem vorab gewählten Kameramodell heraus ergeben, mit möglichst hoher Genauigkeit unter vertretbarem Aufwand zu bestimmen. Diese Parameter wurden aus einer Menge von Leuchtpunkten, deren Lage sich ständig entlang gewisser Trajektorien im Sichtbereich jeder Kamera ändert, und den entsprechenden Projektionen auf der Bildebene der Kameras mit Hilfe eines Optimierungsverfahrens ermittelt. Unter Kenntnis dieser Kameraparameter wird dann die Lage des Objektes aufgrund der visuellen Information von zwei Kameras und 4 Laserdioden in Echtzeit bestimmt.

Zusammenfassend wurden in dieser Arbeit auch die experimentellen Untersuchungen der gesamten Struktur der visuellen Robotersteuerung bei verschiedenen Geschwindigkeiten des Objektes durchgeführt, um experimentell die Robustheit und Stabilität eines solchen Systems nachzuweisen und die maximale Geschwindigkeit der Objektverfolgung zu bestimmen.

Inhaltsverzeichnis

Symbolverzeichnis

Mathematische Bezeichner

p	skalare Größe
$\vec{P}$	vektorielle Größe (Signal)
$P(\cdot)$	vektorielle Funktion (Parametervektor)
P	Matrix

Verzeichnis wichtiger Symbole

Die Symbole sind in alphabetischer Reihenfolge angeordnet. Selten benutzte oder abweichende Bedeutungen werden im Text erläutert.

α	Eulersche Orientierungswinkel (Drehung um die x- Achse)
β	Eulersche Orientierungswinkel (Drehung um die y- Achse)
γ	Eulersche Orientierungswinkel (Drehung um die z- Achse)
$\vec{\theta}$	Gelenkwinkelvektor im Roboterkonfigurationsraum
θ_i	Gelenkwinkel des Roboters
$\vec{\theta}_{ist}$	Istgelenkwinkel des Roboters
$\vec{\theta}_{soll}$	Sollgelenkwinkel des Roboters
λ	Schrittweite des Optimierungsverfahrens
τ	Antriebsdrehmomente
$\Omega(t)$	Winkelgeschwindigkeit
(c_x, c_y)	Verschiebung zwischen Bild- und Sensorkoordinaten
E	Fehlerfunktion
f	Brennweite
H	Hesse- Matrix
h	Abstiegsrichtung
I	Einheitsmatrix
J	Jacobi- Matrix
$L(x)$	Funktion des gemessenen Lichtes
$p(t)$	Verlauf des Positionsvektors des Endeffektors
P	Positionsvektor
$\vec{P}_e = {}^B P_E$	Reale Lage des Endeffektors im Weltkoordinatensystem

$\vec{P}_e^* = {}^B\vec{P}_e^*$	Visuell ermittelte Lage des Endeffektors im Weltkoordinatensystem
$\vec{P}_{ist}$	Istposition und –orientierung des Endeffektors
$p_k=(x_k, y_k, z_k)$	Position des Punktes im Kamerakoordinatensystem
$\vec{P}_{kor}$	Korrekturtrajektorie für die Lage des Endeffektors bei der Objektverfolgung
$\vec{P}_o^*$	Im Primärvorgang grob bestimmte Lage des Zielobjektes
$\Delta\vec{P}_o$	Differenz zwischen der realen und den visuell ermittelten Lage des Endeffektors (bei der Objektverfolgung bildet die Bewegung des Objektes ab)
$\Delta\vec{P}_{soll}$	Gewünschter relativer Abstand zwischen dem Objekt und dem Endeffektor
$\vec{P}_{soll}$	Sollposition und –orientierung des Endeffektors
$\vec{P}_{soll}^*$	Solllage des Endeffektors $\vec{P}_{soll}$ + Korrekturtrajektorie bei der Objektverfolgung $\vec{P}_{kor}$
$p_w=(x_w, y_w, z_w)$	Position des Punktes im Weltkoordinatensystem
R	Rotationsmatrix
S	Summensignal
T	Transformationsmatrix
(t_x, t_y, t_z)	Verschiebung der Position
$v(t)$	Lineare Positionsgeschwindigkeit
x_0	Startpunkt
(x_s, y_s)	Bildkoordinaten
(x_f, y_f)	Sensorkoordinaten

Abbildungsverzeichnis

Kapitel 1

Einleitung

1.1 Motivation

Die Erforschung und Realisierung „sehender“ Maschinen und „intelligenter“ Roboter bildet seit Jahrzehnten einen Arbeitsschwerpunkt der Robotertechnik. Das Ziel dabei ist die Realisierung intelligenter sehender Roboter, um die verschiedenen Aufgaben auszuführen.
Ausgehend von der Hypothese, dass sich die Intelligenz von Lebewesen im Laufe der Evolution aus dem Zusammenwirken von Sehen, Bewegungssteuerung und deren Adaption an die Umwelt entwickelt hat, werden die Grundlagen eines solchen Zusammenwirkens bei autonomen Robotern erforscht. Das Fernziel ist die Realisierung von Robotern mit einer praktischen Intelligenz von der Art, wie sie auch Lebewesen zum Überleben in ihrer jeweiligen Umwelt befähigt.
Höher entwickelte intelligente Lebewesen sind zumeist mit einem Sehsinn ausgestattet, der es ihnen erlaubt, sich in ihrer Umwelt zu orientieren und sich zweckmäßig und zielorientiert zu bewegen, z.B. zur Nahrungssuche oder zur Vermeidung von Gefahren. Nur in den Bereichen, in denen aufgrund des fehlenden Sonnenlichtes keine visuelle Erfassung der Umgebung möglich ist (z.B. in Höhlen oder im Meer), haben sich andere Sensorsysteme als leistungsfähiger erwiesen, beispielsweise die Verwendung von Ultraschall bei Fledermäusen und Delphinen. Deshalb sind die Sichtsysteme zur Entwicklung und Realisierung intelligenter Roboter von entscheidender Bedeutung, da sie es den Robotern ermöglichen, komplexe und dynamisch veränderliche Situationen in Echtzeit zu erkennen. Neben der Fähigkeit, mit Sensoren Situationen in einer dynamischen Umwelt zu erkennen, benötigt ein intelligenter Roboter insbesondere auch die Fähigkeit des Lernens, also die Fähigkeit, durch Interaktion mit der Umwelt Kenntnisse zu erwerben bzw. zu erweitern und Verhaltensweisen zu erlernen bzw. zu verbessern. Es ist zu erwarten, dass Roboter, die eine derartige Intelligenz besitzen, in ihrer jeweiligen Umgebung eine ähnliche Adaptionsfähigkeit aufweisen werden wie Lebewesen.
Gerade in den letzten Jahren gewann das Thema visuelle Objektverfolgung immer mehr an Bedeutung, da viele Anwendungen heutzutage damit verknüpft wurden. Besonders in der Automobilindustrie im Bereich der autonomen Fahrzeugkontrolle und der Fahrerunterstützung kommt Bildverarbeitung in Kombination mit Objektverfolgung zum Einsatz. Aber auch in anderen Gebieten wie der Medizin sind Objektverfolgungsszenarien unverzichtbar geworden. Auf diese Beispiele wird in dieser Arbeit Bezug genommen, sie sind letztendlich als praktisches Anwendungsfeld für Ergebnisse im wissenschaftlichen Umfeld anzusehen.

1.2 Allgemeine Aufgabenstellung

Die Roboter finden weiterhin zunehmende Anwendung in vielen Bereichen der Industrie, um Prozessabläufe bei Fertigung, Vertrieb und Logistik (z.B. bei Sortierung, Demontage oder Montagevorgängen sowie bei der Objektverfolgung) zu optimieren. Ausgehend von der Aufgabe, fest vorgegebene Arbeitsschritte mit hoher Präzision und Geschwindigkeit durchzuführen, besteht ein Ziel angewandter Forschung darin, die Einsatzmöglichkeiten derart zu erweitern, dass Roboter auch auf sich ändernde Umgebungsbedingungen flexibel reagieren können. Dazu ist es erforderlich, einen Roboter mit einer Sensorik auszustatten, die es ihm ermöglicht, seine Umgebung und damit auch Änderungen in dieser Umgebung wahrnehmen zu können.
Optische Sensorik bietet für einen solchen Einsatz den Vorteil, dass zur Ermittlung der Umgebungsparameter, ähnlich wie bei Mensch und Tier, die visuelle Wahrnehmung in Form von Bildern ausgewertet wird. Diese Bilder können nicht nur für die Berechnung der Umgebungsparameter und damit zur Steuerung und Navigation des Roboters, sondern zusätzlich auch zur Überwachung der Robotertätigkeit verwendet werden.
Die gegenwärtige Entwicklung des Industrieroboters zu einem Massenprodukt äußert sich in einem immer weiter sinkenden Anteil der einmalig für Projektierungs- und Entwicklungsleistungen sowie für die zu installierende Technik aufgewendeten Kosten, wohingegen der sich oftmals wiederholende Aufwand für eine erneute Programmierung und Konfiguration in einer (u.U. nur unerheblich) gewandelten Applikation weitgehend unverändert bleibt. Dieser Entwicklung soll eine Perspektive gegenübergestellt werden, welche die mittlerweile erheblichen rechentechnischen Möglichkeiten ausnutzt, die im Rahmen der aus dem Preisverfall frei werdenden Mittel verfügbar werden. Hiermit lässt sich ein grundlegender Ausbau der Steuerungsfunktionalitäten realisieren, mit denen der Aufwand zur Anpassung und Rekonfiguration für geänderte Manipulationsaufgaben ebenfalls drastisch reduziert werden kann. Das Endziel einer derartigen Entwicklung besteht darin, dass sich der Roboter mittels multisensorieller Informationen (taktil, visuell usw.) permanent an veränderliche Aufgaben und Umgebungsbedingungen adaptiert und nach außen keine zustandsabhängigen Eigenschaften aufweist. Für dieses Leistungsmerkmal soll hier der Begriff der *Arbeitsraumautonomie* geprägt werden. Auf der Basis einer solchen Entwicklung wird die bisherige Bindung an einen definierten unveränderlichen Umgebungszustand aufgehoben. Damit ließe sich auch die methodische Abtrennung vom Entwicklungszweig der *Servicerobotik* überwinden, die grundsätzlich von umgebungsadaptiven Steuerungseigenschaften ausgehen muss.
Im Mittelpunkt eines solchen Steuerungskonzeptes stehen verschiedene Formen von intern anzulegenden Modellen, die permanent den tatsächlichen Zuständen nachgeführt werden. Diese Modelle lassen sich in 3 Ebenen strukturieren:

1. Die Ebene des Umweltmodells ermöglicht es der Robotersteuerung, aus den ihr zur Verfügung stehenden sensorischen Informationen und Merkmalen eine aufgabengerechte Bewegung des Endeffektors im Raum zu erzeugen. Die (mit dem Roboter interagierenden) Objekte stellen hierbei Elemente dieser Umwelt dar.
2. Die Ebene eines adaptiven Kinematikmodells dient der Transformation von räumlichen Trajektorien in entsprechende Bewegungen der einzelnen Robotergelenke, womit auch geometrische Veränderungen der Roboterstruktur selbst

berücksichtigt werden.

3. In der untersten Ebene eines dynamischen Modells erfolgt schließlich die Realisierung der Achsbewegungen durch Erzeugung der erforderlichen Antriebskräfte und -momente.

Als Schnittstelle zwischen den Ebenen treten somit die kartesische Beschreibung der Bewegungstrajektorien einerseits, und deren Transformation in den Konfigurationsraum der Gelenkwinkel andererseits auf. In dieser Arbeit wird aber die letzte Ebene nicht betrachtet, sondern es wird ein für höhere dynamische Anforderungen bereits geeigneter Roboter ausgewählt. Die Ebenen 1 und 2 werden hier als Gesamtheit betrachtet. Damit ergeben sich aus den zur Anwendung kommenden Technologien eine Reihe weiterer Fragestellungen, die im Rahmen dieser Arbeit zu erörtern sind. Diese gliedern sich in die folgenden drei Schwerpunkte:

- ***Trajektoriengenerierung***
  Bei der Interaktion von Robotern mit ihrer Umwelt spielt die Planung ihrer Bewegungsabläufe in der jeweiligen Umgebung eine herausragende Rolle. Da insbesondere industrielle Roboter möglichst selbstständig einen gewünschten Zielort oder eine bestimmte Zielkonfiguration erreichen sollen, gehört die Bewegungsplanung zu ihren wesentlichen Aufgaben.

- ***Berechnung der Kinematik***
  Es ist erforderlich, die kartesische Beschreibung des Trajektorienverlaufes des Endeffektors in den Achskoordinaten des Roboters und umgekehrt zu bestimmen.

- ***Berechnung der Lage der Objekten aus den zur Verfügung stehenden sensorischen Informationen***
  Um die Lage des bewegten Objektes im Arbeitsraum des Roboters zu bestimmen, muss der physikalische Raum mit Hilfe der Sensoren ständig neu erfasst werden können, sodass eine möglichst exakte Berechnung der für die weitere Fortbewegung des Roboters nötigen Informationen erfolgen kann. Unter der Lage des Objektes versteht man hier seine Position und Orientierung bezüglich des Objekts, mit dem der Roboter interagiert. Um diese zu bestimmen, müssen mehrere Sensoren vorhanden sein, mit denen sich diese Relation erfassen lässt.

Das gesamte optisch geführte Steuerungskonzept eines Roboters ist in unserem Fall zur Lösung folgender Aufgaben vorgesehen:

- Annäherung des Endeffektors an das Objekt, das sich an beliebiger Position im Arbeitsraum befindet, um damit einen Greifvorgang zu ermöglichen.

- Verfolgung des bewegten Objektes innerhalb des Arbeitsraumes des Roboters, um an beliebiger Position im Arbeitsraum einen Greifvorgang einleiten zu können.

1.3 Überblick

Ausgehend von Erkenntnissen und aktuellen Entwicklungen in der Robotik und Sensorik werden Untersuchungen für die optisch geführte Steuerung von Robotern im Rahmen dieser

Arbeit durchgeführt. Die vorliegende Arbeit ist in sieben Kapitel unterteilt und ist in starker Anlehnung an die oben formulierte Aufgabenstellung strukturiert.

Nachdem in diesem Kapitel eine Motivation zum Aufbau der visuellen Robotersteuerung und eine allgemeine Aufgabenstellung gegeben wurden, beschreibt das Kapitel 2 sowohl den Überblick und die Literaturauswertung zur visuellen Robotersteuerung als auch die Entwicklung einer Struktur der optisch geführten Robotersteuerung.

Im Kapitel 3 wird die Hardwarerealisierung des Systems zur praktischen Verifizierung dargestellt. Die Ausrichtung dieser Arbeit wird von Anfang an wesentlich durch den Grundsatz bestimmt, dass alle Forschungsergebnisse zum Nachweis ihrer Gültigkeit in praxisnahen Experimenten in der realen Umwelt überprüft und demonstriert werden. Diese Arbeitsweise ist aufwendig, hat aber gegenüber reinen Simulationen den gravierenden Vorteil, zu weitaus solideren und in der Praxis belastbareren Ergebnissen zu führen. Grundvoraussetzung für diese Arbeit sind daher Roboter und ein leistungsfähiges Echtzeitberechnungssystem. Die Kopplung dieser beiden Teileinheiten zu einem System, dass zur Verifikation des entstehenden Steuerungskonzeptes geeignet ist, wurde bereits zu Beginn der Arbeit realisiert.

Kapitel 4 widmet sich der Trajektoriengenerierung im kartesischen Arbeitsraum des Roboters. Bei vielen Anwendungen ist eine ruhige, stetige Bewegung des Roboters entlang einer Reihe von Punkten, die seine Trajektorie definieren, erforderlich. In diesem Kapitel werden die Methoden zur Lösung dieses Problems erörtert.

Kapitel 5 liefert die Grundkenntnisse der Kinematik des Roboters. Die Kinematik beschreibt die geometrische Bewegung eines mechanischen Systems. Bei Robotern geht es dabei insbesondere um die Beziehung zwischen den Angaben der Winkelkoordinaten für die Antriebsachsen und den kartesischen Koordinaten speziell des Endeffektors. Dabei unterscheidet man zwischen der direkten und der inversen Kinematik. Die direkte Kinematik ordnet einer bestimmten Stellung seiner Gelenke, d.h. einer Konfiguration, eine Position und Orientierung des Endeffektors im kartesischen Arbeitsraum zu. Die Umkehrung dieses Zusammenhangs beinhaltet die inverse Kinematik. Sie ordnet der Position und Orientierung des Endeffektors im kartesischen Arbeitsraum die Konfigurationen des Roboters zu und ist für serielle Kinematiken mathematisch wesentlich komplexer. In diesem Kapitel wird die Vorgehensweise zur Berechnung der direkten Kinematik erörtert. Da die inverse Kinematik ein sehr komplexes Problem wegen ihrer starken Nichtlinearität und Mehrdeutigkeit darstellt, wird sie mittels eines speziell angepassten Levenberg-Marquardt Optimierungsverfahrens berechnet. Anschließend werden die Ergebnisse der simulativen und experimentellen Untersuchungen der Kinematik auch unter Betrachtung der Genauigkeit der Berechnungen dargestellt.

Kapitel 6 beschreibt die Grundlagen des Aufbaues eines Kameramodells und die Vorgehensweise bei der Kamerakalibrierung. Wenn in einer visuellen Steuerung zur Erfassung des Objektes Kameras eingesetzt werden, dann müssen die verwendeten Kameras kalibriert werden, um Korrespondenz zwischen Punkten auf den zweidimensionalen Bildern und den Punkten der darin dargestellten Objekte in der dreidimensionalen Welt herzustellen. Dabei gilt es, die inneren und äußeren Kameraparameter, die sich aus einem vorab gewählten Kameramodell heraus ergeben, mit möglichst hoher Genauigkeit unter vertretbarem Aufwand zu bestimmen. Innere Parameter umfassen dabei die internen geometrischen und optischen Charakteristika der Kamera, die äußeren Parameter beschreiben die 3D- Position und -

Orientierung der Kamera relativ zur betrachteten Welt. Sind die Kameraparameter ermittelt, kann die Lage des Objektes entsprechend der zugrunde liegenden Aufnahmen des Objektes bestimmt werden.

Die Ergebnisse dieser Arbeit sowie die experimentelle Verifikationen der gesamten optisch geführten Robotersteuerung werden anschließend im Kapitel 7 zusammengefasst. Dabei werden die Genauigkeit der Lagebestimmung des Objektes und die notwendige Werte der Schranke zur Unterdrückung der optischen Störungen ermittelt. Um experimentell zu testen, wie der Roboter das Objekt verfolgt, wird ein Laufband mit Hilfe einer SPS- Steuerung für die Bewegung des Objektes mit verschiedenen Geschwindigkeiten programmiert. Dabei wurde eine Stabilitäts- und Robustheitsanalyse der vorgeschlagenen Struktur der Robotersteuerung durchgeführt. Es wurde auch die maximal mögliche Geschwindigkeit der Objektverfolgung bestimmt.

Im Anhang A wird der Aufbau der Software zur Realisierung der optisch geführten Steuerung im Signalprozessorsystem beschrieben. Diese Steuerung wurde basierend auf einen DSpace-Mehrprozessorsystem und einem 7-achsigen wissenschaftlichen Roboter „ *amtec r7* “ aufgebaut. Aufgrund des Mehrprozessorensystems zur Realisierung der optisch geführten Robotersteuerung in Echtzeit wurde entsprechende Software entwickelt, die auf verschiedenen Prozessoren parallel arbeitet.

Kapitel 2

Entwicklung der Struktur der optisch geführten Robotersteuerung

2.1 Überblick und Literaturauswertung zur visuellen Robotersteuerung

Flexible Produktions- und Fertigungssysteme erfordern den Einsatz intelligenter Roboter, die Gegenstand intensiver Forschung sind [30, 34, 56]. Diese sind mit entsprechenden Sensoren auszurüsten, um Aufgaben unter Bedingungen, die vorher nicht vollständig bekannt sind bzw. sich im Laufe der Zeit verändern, lösen zu können.
Um von der in konventionellen Systemen weit verbreiteten Festprogrammierung des Roboters, die z.B. durch sogenanntes „Teach-in" realisiert wird, zu einer flexibleren Gestaltung des Fertigungsprozesses zu gelangen, ist also der Einsatz entsprechenden Sensoren zur Erfassung der Umgebung des Roboters erforderlich. Beispiele dafür, dass der Roboter durch Auswertung von Sensorinformationen in einer relativ unbekannten Umgebung arbeiten kann, zeigen [7, 65, 72].
Zur Bestimmung der Position und Orientierung von Zielobjekten werden Geräte oder Gerätesysteme eingesetzt, die man als Positionsmesssysteme oder kurz Tracker bezeichnet. Die Tracker stellen einen entscheidenden Bestandteil eines Systems der sensorgeführten Robotersteuerung dar, da sie es ermöglichen, eine Korrelation zwischen der Lage des Objektes und den Messdaten der Sensoren herzustellen. In den vergangenen 35 Jahren wurden verschiedene Typen von Trackern entwickelt. Diese können anhand des für die Messung zugrunde liegenden physikalischen Prinzips grob in drei Kategorien eingeteilt werden: magnetische, akustische und optische Tracker. Jede dieser Klassen besitzt ihre Vorteile, Nachteile und spezifische Anwendungsbereiche. Einen Überblick über die verschiedenen Tracker gibt Allen in [2].
Um die Leistung der verschiedenen heute verfügbaren Tracker zu erfassen und zu vergleichen, sind unter anderem die folgenden Kriterien geeignet:

- **Statische Genauigkeit** (Static accuracy): Dieses Maß gibt die maximale Abweichung des gemessenen Positions- und Orientierungswertes eines Referenzpunktes vom tatsächlichen Wert an.
- **Dynamische Genauigkeit** (Dynamic accuracy): Die dynamische Genauigkeit gibt die Genauigkeit des Systems bei Bewegung des Sensors mit verschiedenen Geschwindigkeiten an.

- **Auflösung** (Resolution): Mit der Auflösung gibt man die kleinste Änderung in den Positions- und Orientierungswerten an, die noch vom Tracker detektiert werden kann.
- **Wiederholrate** (Update rate): Diese Angabe betrifft die Rate, mit der jeweils neue Messwerte an den Hostrechner übertragen werden.
- **Latenzzeit** (Phase lag / Latency): Die Totzeit zwischen einer Veränderung der Position und Orientierung und der Übertragung der Änderung an den Hostrechner wird mit Latenzzeit bezeichnet. Dieses Maß umfasst sowohl die eigentliche Latenzzeit des Meßsystems als auch die Aktualisierungsrate im Hostrechner.
- **Arbeitsvolumen** (Working volume): Dieses Maß beschreibt die Ausdehnung des Raumes, innerhalb dessen der Tracker die Messwerte mit der spezifizierten Genauigkeit und Auflösung aufnehmen kann.
- **Zahl unabhängiger Sensoren**: Bei der Bestimmung der Lage mehrerer Objekte muss die unabhängige Messung durch mehrere Sensoren möglich sein. Die Sensoren sollten sich nicht gegenseitig beeinflussen und ihre Zahl der Wiederholrate möglichst nicht senken.
- **Zahl der Freiheitsgrade** (Degrees of freedom, DOF): Diese Angabe bezieht sich auf die Anzahl der Raumrichtungen, die bei der Messung frei variieren können. Maximal existieren drei Positionsrichtungen und drei Orientierungen, also insgesamt sechs Freiheitsgrade.

Ein guter Tracker sollte eine hohe Genauigkeit, eine feine Auflösung und eine hohe Wiederholrate besitzen. Die Latenzzeit sollte gering und das Arbeitsvolumen möglichst groß sein. Im Idealfall sollte er keine spezielle Umgebung für die Arbeit benötigen. Außerdem sollten diejenigen Teile des Trackers, die an beweglichen Objekten befestigt werden müssen, klein und leicht sein. Ein moderner Tracker deckt in der Regel alle sechs Freiheitsgrade ab.
Im folgenden werden drei grundlegenden Bauarten von Trackingsystemen vorgestellt.

Akustische Tracker

Akustische Tracker nutzen Ultraschallsignale, um die Position von Zielobjekten zu bestimmen. Sie können in zwei Klassen eingeteilt werden, die Phasenkohärenztracker (PK) und die „Time-of-flight"- Tracker (TOF).

PK- Tracker

Phasenkohärenztracker bestimmen Position und Orientierung, indem sie die Phase von emittierten akustischen Wellen mit der Phase einer Referenzwelle vergleichen. Die Sender werden am Zielobjekt montiert, die Empfänger befinden sich an bekannten fixen Positionen in der Trackingumgebung. Die Empfänger messen periodisch die Phasendifferenz zwischen den emittierten Wellen und einer Referenzwelle. Da ein Phasenwinkel von 360° äquivalent zu einer Wellenlänge ist, kann die Differenz zwischen zwei aufeinander folgenden Messungen in die Distanz umgerechnet werden, die der Sender zwischen den beiden Messungen zurückgelegt hat, vorausgesetzt, diese Distanz ist geringer als eine Wellenlänge. Um diese Bedingung einzuhalten, müssen die Empfänger in der Lage sein, die Messungen der Phasendifferenz ausreichend schnell auszuführen. Die relativen Bewegungen des Zielobjekts werden genutzt, um dessen Position und Orientierung zu aktualisieren. Das System wird durch Messung der Position und Orientierung durch eine externe Quelle initialisiert. Da PK- Tracker jeweils die vorhergehende Position und Orientierung und die

ermittelte Veränderungen in die Berechnung einbeziehen, akkumulieren sich die Fehler mit der Zeit und führen zu Ungenauigkeiten in der ermittelten Position und Orientierung. Daher müssen derartige Tracker von Zeit zu Zeit durch eine externe Referenz korrigiert werden. Diese Tracker sind in den 60er Jahren konzipiert und eingesetzt worden, in neuerer Zeit ist kein Tracker dieser Art für ein Steuerungssystem verwendet worden.

TOF- Tracker

TOF- Tracker berechnen die Position und Orientierung des Zielobjekts, indem sie die Zeit messen, die Ultraschallimpulse benötigen, um von einer Gruppe von Sendern zu einer Gruppe von Empfängern zu gelangen. Ein typisches System besteht aus einem am Zielobjekt montierten Stab mit drei Sendern und aus vier Mikrofonen als Empfänger an fixen Positionen mit bekannter geometrischer Anordnung in der Trackingumgebung. Die Sender emittieren Ultraschallimpulse in regelmäßigen Intervallen, wobei jeweils ein unterschiedlicher Sender zur gleichen Zeit emittiert. Die Empfänger fangen das Signal auf und messen die Ankunftszeit.
Da der Zeitpunkt der Emission bekannt ist, lässt sich aus dieser Messung die Laufzeit des Impulses berechnen. Durch Multiplikation der Laufzeit mit der Schallgeschwindigkeit in der Luft wird der Abstand zwischen Sender und Empfänger bestimmt. Diese Abstandswerte und die bekannten Positionen der Empfänger bieten ausreichend Information, um mittels Triangulierung die Position und Orientierung des Zielobjektes zu berechnen.
Eines der Hauptprobleme mit TOF- Tracker ist deren begrenzte Update-Rate, die durch die geringe Geschwindigkeit des Schalls in der Luft bedingt ist. In [14] wird von erreichbaren 15-20 Messungen pro Sekunde berichtet. Ein zusätzliches Problem ist durch die Veränderung der Schallgeschwindigkeit bei Veränderung der Luftfeuchtigkeit, Lufttemperatur, Luftdruck und bei Turbulenzen gegeben. Dieses Problem kann allerdings durch permanente Messung der Schallgeschwindigkeit mit einem gesonderten Sender-Empfänger-Paar überwunden werden, sofern man davon ausgehen kann, dass die Schallgeschwindigkeit im gesamten Arbeitsvolumen konstant ist.

Vorteile akustischer Tracker

- Sie sind klein und leicht und daher für bewegliche Objekte geeignet.
- Sie werden im Gegensatz zu den elektromagnetischen Trackern nicht durch Magnetfelder gestört.
- Sie erfordern keine speziell angepasste Umgebung.

Nachteile akustischer Sensoren

- Hindernisse zwischen Sender und Empfänger beeinträchtigen die Funktion.
- Echos und externes Rauschen können Fehlmessungen und damit Ungenauigkeiten hervorrufen.
- TOF- Tracker haben geringe Wiederholraten.
- Die von den aktuellen Parametern des Mediums Luft abhängige Schallgeschwindigkeit macht eine kontinuierliche Referenzmessung notwendig.

Optische Tracker

Optische Tracker wurden unter Ausnutzung einer ganzen Reihe von technologischen Prinzipien entwickelt. Man unterscheidet bei den optischen Tracker die Klasse der

„Beacon"- Tracker (Leuchtfeuer-, Leitsignal- Tracker) und die Klasse der sonstigen Typen. „Beacon"- Tracker basieren auf der Detektion passiver oder aktiver Marker und werden daher auch als Marker- Tracker bezeichnet. Besonders hervorgehoben unter den sonstigen Typen sind die Laser-Range- Tracker zu nennen, deren Arbeitsprinzip auf der Messung des Kontrastverhältnisses des von einem Beugungsgitter bei Laserbestrahlung auf dem Objekt erzeugten Beugungsmusters beruht. Da diese Bauform für heutige Steuerungssysteme nicht zum Einsatz kommt, wird im folgenden nur auf die markerbasierten Tracker eingegangen.
Die bei Marker- Tracker zum Einsatz kommenden Marker können entweder aktiv, also Licht aussendend, oder passiv, also Licht reflektierend, sein. Um die Marker zu detektieren, werden in der Regel mehrere Sensoren, wie beispielsweise Kameras oder Lateraleffekt-Fotodioden verwendet. Abhängig von der Position der Signalelemente und der Sensoren kann man die Messsysteme in zwei Kategorien einteilen: *Inside-out-* und *Outside-in-* Systeme. Bei *Inside-out-* Systemen werden die Signalelemente an festen Positionen in der Umgebung und die Sensoren am Zielobjekt platziert, bei *Outside-in-* Systemen ist es umgekehrt. Beide Technologien besitzen ihre Vorteile. Ein *Inside-out-* System kann die Orientierung des Zielobjekts mit größerer Genauigkeit bestimmen, als ein *Outside-in-* System mit vergleichbarer Technologie. Allerdings besitzen *Inside-out-* Systeme den Nachteil, dass die Lage des Objektes schwer zu bestimmen ist, wenn es so platziert ist, dass nur wenige Marker von den Sensoren aus sichtbar sind. Außerdem sind die Sensoren recht groß und schwer und daher in vielen Anwendungen nicht praktikabel.

Inside-out-Systeme

Ein gutes Beispiel für ein *Inside-out-* System ist der „Ceiling-Tracker" der University of North Carolina. Dieses System nutzt Infrarot-Leuchtdioden (LED) als Signalelemente, die in großer Menge an festen Positionen der Decke eines großen Raumes angebracht wurden. Vier Lateraleffekt- Fotodioden und deren Linsen dienen als Sensoren und werden am Zielobjekt montiert. Im laufenden Betrieb werden die LEDs im Bereich des Sichtfeldes der Sensoren nacheinander für einen kurzen Moment durch den Hostrechner angesteuert. Die Linsen der Sensoren erzeugen Bilder der LEDs auf den Lateraleffekt-Fotodioden, die ihrerseits die 2-D- Position dieser Bilder messen. Jede gemessene 2-D- Position definiert einen 3-D-Vektor von einem Sensor zu einer sichtbaren LED. Aus den Vektoren und den bekannten Positionen der LEDs werden Position und Orientierung des Zielobjekts mit Hilfe einer photogrammetrischen Technik berechnet. Durch die Montage an der Decke lässt sich ein sehr großes Arbeitsvolumen erzielen.

Outside-in-Systeme

Bei den aktiven *Outside-in-* Systemen werden an dem zu untersuchenden Objekt mehrere Infrarot- LEDs in definierter Anordnung angebracht, die sequentiell zu definierten Zeiten angesteuert werden. Das Objekt wird mit der an einem festen Ort montierten Infrarotkamera beobachtet. Jede der nacheinander aufleuchtenden LEDs erzeugt ein Abbild auf der Bildebene der Kamera, welche die 2-D-Position dieses Abbildes misst. Jede dieser gemessenen 2-D Abbildungen definiert einen Vektor von einer LED zur Kamera. Aus diesen Vektoren lassen sich die Positionsparameter des Objektes berechnen.
Bei passiven *Outside-in-* Systemen werden auf den Zielobjekten optisch detektierbare Marker angebracht. Diese Marker werden mit Infrarotlicht beleuchtet und von Kameras aufgenommen. Mittels Mustererkennungsalgorithmen werden die Abbilder der Marker aus den Kamerabildern erkannt und den Markern zugeordnet. Aus den 2-D-Positionen können wie bei den aktiven Systemen die Positionsparameter des Objekts berechnet werden. Dieses

Verfahren hat den Vorteil, dass am zu untersuchenden Objekt keine aktiven Bauelemente befestigt werden müssen und keine Kabelverbindungen nötig sind. Bei einem videometrischen System der Firma Honeywell sind die Marker durch reflektierende Muster realisiert, bei dem ProReflex- System der Firma Qualisys dienen einfache punktförmige Reflektoren als Marker.

Vorteile optischer Tracker

- Optische Tracker haben hohe Wiederholraten, die nahezu ausschließlich durch die Prozessorleistung für die Berechnungen begrenzt sind.
- Das Arbeitsvolumen variiert von System zu System, kann aber sehr groß sein.
- Optische Tracker werden nicht durch Metallkörper gestört.
- Passive Reflektoren können kabellos verwendet werden.
- Die Marker können sehr klein und leicht gestaltet werden.

Nachteile optischer Tracker

- Alle optischen Tracker benötigen eine freie Sichtlinie. Ihre Leistung sinkt, wenn es Hindernisse zwischen einem Sensor und den Signalgebern gibt. Dieses Problem kann teilweise durch Einsatz einer Vielzahl von Sensoren und Signalelementen behoben werden.
- Die Leistung optischer Tracker wird durch Hintergrundbeleuchtung und Infrarotstrahlung negativ beeinflusst. Dies muss beim Design der Umgebung berücksichtigt werden.
- Auch wenn die Marker kompakt konstruiert werden können, werden zur vollständigen Erfassung dreidimensionaler Objekte mehrere im Raum verteilte Marker benötigt.

Diverse moderne aktive und passive optische Trackingsysteme wurden hinsichtlich ihrer Genauigkeit untersucht und für den Einsatz als geeignet befunden. Aufgrund der guten Genauigkeitseigenschaften nutzt die Mehrzahl der Steuerungssysteme diese Trackingtechnologie trotz der genannten Nachteile.

Elektromagnetische Tracker

Zwei Typen von elektromagnetischen Trackern mit unterschiedlichen Eigenschaften werden unterschieden, je nachdem, ob ihr Messprinzip auf Wechselfeldern (AC) oder Gleichfeldern (DC) beruht. Beide Typen besitzen jeweils eine Quelle und einen oder mehrere Empfänger.

AC-Tracker

Die Quelle eines AC- Trackers enthält drei wechselseitig senkrecht angeordnete elektromagnetische Spulen und wird an einer festen Position im Raum fixiert. Der Empfänger ist analog konstruiert und wird am Zielobjekt, welches getrackt werden soll, angebracht. Werden die drei Spulen der Quelle von Wechselstrom durchflossen, bildet sich ein magnetisches Wechselfeld aus. Dieses Feld induziert Ströme in den passiven Empfängerspulen. Frequenz, Phase und Amplitude der induzierten Ströme verändern sich als Funktion der Position und Orientierung des Empfängers in Bezug auf die Quelle. Sie werden gemessen und daraus die Position und Orientierung des Empfängers berechnet. In der Gegenwart metallischer Objekte sind AC- Tracker vom Problem der Störungen durch Kriechströme betroffen. Gemäß dem Faradayschen Gesetz induziert das magnetische

Wechselfeld Kriechströme in metallischen Objekten in der Umgebung. Die Kriechströme erzeugen ihrerseits ihre eigenen magnetischen Felder, welche das magnetische Feld der Quelle überlagern und es stören. Als Folge dieser Störung gibt der Tracker ungenaue Positions- und Orientierungswerte in der Nähe von Metallobjekten aus. AC-Tracker können daher nicht in allen Umgebungen eingesetzt werden.

DC-Tracker

Bei DC- Trackern werden Störungen durch Kriechströme in der Nähe von Metallobjekten vermieden, indem ein statisches Magnetfeld genutzt wird. Die Quelle des DC- Trackers wird mit kurzen Gleichstromimpulsen angeregt. Nachdem die sich durch die steigende Flanke ergebenden Perturbationen ausgeklungen sind, bleibt das durch den Transmitter erzeugte Magnetfeld statisch bis zum Ende des Impulses. Die Feldstärke des statischen Feldes wird durch den Empfänger (beispielsweise durch Fluxgate- Sensoren) gemessen. Die Messung enthält allerdings einen beträchtlichen Fehleranteil, der durch das Erdmagnetfeld hervorgerufen wird. Um diese unerwünschte Komponente zu beseitigen, wird das Erdmagnetfeld gemessen, bevor der Transmitter angeregt wird. Diese Messung wird von der folgenden Messung des übertragenen Feldes subtrahiert und das Ergebnis für die Berechnung von Position und Orientierung des Zielobjektes verwendet. Zwar umgehen DC-Tracker die Störungen, die durch Kriechströme in Metallen hervorgerufen werden, sie bleiben jedoch empfindlich gegenüber den Einflüssen ferromagnetischer Materialien, die in der näheren Umgebung des Trackers das übertragene magnetische Feld durch Reflexionen stören, und gegenüber externen magnetischen Störquellen, wie Netzgeräte oder Bildschirme.

Vorteile magnetischer Tracker

- Die Sensoren magnetischer Tracker sind klein und leicht.
- Magnetische Tracker benötigen keine freie Sichtlinie zwischen Transmitter und Empfänger. Optische Hindernisse bedeuten keine Einschränkung der Funktion.
- Magnetische Tracker benötigen nur einen Sender und einen Empfänger pro Zielobjekt.
- Magnetische Tracker besitzen eine ausreichende Updaterate.
- Sie sind preisgünstig erhältlich.

Nachteile magnetischer Tracker

- Hauptproblem sind die Störungen durch Metalle, ferromagnetische Stoffe oder elektromagnetische Störfelder im Arbeitsvolumen.
- Für ein großes Arbeitsvolumen sind hohe Feldstärken notwendig, die große Spulen erfordern und Störquellen für andere Geräte sein können.

Wegen oben erwähnter Vorteile wurden in der Regel optische Tracker (Sensormesssysteme) zur sensorgeführten Robotersteuerung ausgewählt. Zur Erfassung der Lage des Objektes lassen sich außerdem noch verschiedene optische Sensoren in einem Robotersystem einsetzen. Dabei werden CCD-Kameras häufig untersucht und verwendet. Für die Aufgaben, bei denen sich Kontakt zwischen dem Roboter und dem Werkstück ergibt, werden im allgemeinen die Kraft-Moment- Sensoren verwendet [7, 71]. Bei der freien Bewegung sind die optischen (visuellen) Sensoren gut geeignet für nahezu alle weiteren Aufgaben. Als ein wichtiger Sensor in der Robotik ist die Kamera von großer Bedeutung,

weil sie analog zum menschlichen Auge bei der nichtkontaktbehafteten Messung der Umgebung eine wesentliche Rolle spielt [35, 40].
Für die visuelle Steuerung liefert [19, 20] einen Überblick der bisherigen Untersuchungen zur Robotersteuerung mittels visueller Informationen. Beim Einsatz von optischen Sensoren ist die visuelle Steuerung eine Grundproblematik in der Robotik. Dazu sind die nichtlinearen Relationen zwischen visueller Information und Achswinkeln zu untersuchen.
In den herkömmlichen Verfahren sind die folgenden Schritte für die Robotersteuerung mittels der visuellen Information notwendig:

Erfassung und Bearbeitung der visuellen Information

Für den Einsatz der visuellen Information in der Robotersteuerung müssen zuerst die entsprechenden visuellen Daten erfasst und weiter verarbeitet werden. Die visuelle Information der zu bearbeitenden Objekte kann durch CCD-Kameras aufgenommen werden. Die aufgenommenen Bilder werden durch das verwendete Bildverarbeitungssystem digitalisiert und gespeichert. Auf Basis dieser Bilddaten werden die notwendigen Merkmale (Kameraparameter) bei Verwendung entsprechender Algorithmen (Kalibrierung) extrahiert. Für die räumlichen Merkmale muss eine weitere Verarbeitung mittels der Zuordnungsalgorithmen sowie 3D- Verarbeitungsmethoden durchgeführt werden. Durch die Erfassung und Bearbeitung der Sensorinformation lassen sich die nach Verarbeitung erhaltenen Daten direkt in der Robotersteuerung verwenden.

Ermittlung der Robotersteuerung mittels visueller Information

Die durch Bildverarbeitung ermittelten visuellen Merkmale, die relativ zum Kamerakoordinatensystem sind, werden zum Basiskoordinatensystem transformiert. Dabei basiert die Transformation auf der Kalibrierung des Hand-Augen-Systems. Ferner wird die räumliche Zielkonfiguration (Position und Orientierung des Endeffektors) anhand der im Basiskoordinatensystem dargestellten visuellen Information berechnet. Aufgrund dieser räumlichen Zielkonfiguration des Endeffektors werden die Zielgelenkwinkel des Roboters durch die inverse Kinematik geliefert. Mittels der Ist- und der Zielkonfiguration des Roboters werden die entsprechenden Führungsgrößen ermittelt.
Herkömmlich sind Kameramodelle und die inverse Kinematik des Roboters für eine visuelle Robotersteuerung erforderlich [2, 36]. Ausgehend von der Diskussion im Kapitel 6 müssen hierbei äußere und innere Kameraparameter ermittelt werden, um ein Gesamtkameramodell zu gewinnen. Deshalb ist ein Kalibrierungsvorgang erforderlich.

Visuelle Steuerung:

Bisherige Untersuchungen und Anwendungen zur visuellen Steuerung werden im Folgenden chronologisch vom Anfang der siebziger Jahre bis zum aktuellen Stand diskutiert. Die beschriebenen Anwendungen sind von verschiedenster Art, z.B. in der Fertigung, Teleoperation, Raketenverfolgung, Auswahl von Früchten. Wegen der technischen Begrenzungen, sind die frühen Arbeiten als „*look-then-move*“ Robotersteuerung ausgelegt. Bis Ende der 70er Jahre hatten Systeme gezeigt, dass sie mit einer Abtastfrequenz von 10Hz zur Positionsteuerung, für die Verfolgung, Schweißarbeiten und das Greifen von den bewegbaren Objekten realisierbar sind.
Eine der frühesten Untersuchungen ist die Arbeit von Shirai und Inoue [82] 1973, die beschreibt, wie eine visuelle Steuerung in der Positionssteuerung der Roboter verwendet und die Genauigkeit der Ausführung verbessert werden kann. Das untersuchte System

ermöglicht, dass der Roboter ein Prisma aufgrund visueller Steuerung greifen und in einen Kasten stellen kann. Kantenextraktion und Linienverbesserung werden zur Ermittlung der Konfiguration des Kastens verwendet. Dabei war die Kamera festpositioniert, die Zykluszeit betrug 10s.
In [96] wird ein monokulares Hand-Augensystem für die Positionierung des Endeffektors in einer Ebene mittels visueller Information diskutiert. Dazu sind die Kalibrierung des Hand-Augensystems und die Berechnung der entsprechenden Transformationen erforderlich. In [2] wird ein Hand-Augensystem zum Verfolgen und Greifen eines beweglichen Objektes entwickelt, das sich auf einer bestimmten Bahn mit bestimmter Geschwindigkeit bewegt. Dabei wurden zwei Kameras zur Gesamtbahnbeobachtung festpositioniert, die entsprechend kalibriert sein müssen. Bei diesen Anwendungen mit herkömmlichen Hand-Augen-Systemen ist die Qualität der meist sehr aufwendigen Kalibrierung entscheidend für die Genauigkeit. Daher wurden viele Untersuchungen für eine effektive Kalibrierung durchgeführt [52, 54, 55, 80, 90].
In einem anderen Beitrag stellen Hager, Chund und Morse [36] ein Verfahren für die Positionierung mit einem Hand-Augen-System und dessen Erprobung vor. Dabei wird die Abweichung zwischen dem Zielobjekt und dem Endeffektor mittels visueller Information definiert und diese Information zur Steuerung des Roboters verwendet. Hervorzuheben ist dabei, dass die Positioniergenauigkeit nicht mehr so stark wie bei den oben erwähnten Systemen von der Kalibrierungsgenauigkeit abhängt. Allerdings nimmt der Berechnungsaufwand zu, bedingt durch die notwendige Berechnung der Roboterkinematik und der zugehörigen Hand-Augen-Transformation. Letztere begrenzt die Anpassungsfähigkeit des Gesamtsystems.
In den oben genannten Beispielen sind messtechnisch und zeitlich aufwendige Kalibrierungsverfahren zur Bestimmung der Kameraparameter und der Relationen zwischen Kamera- und Roboterkoordinatensystem erforderlich [36, 40, 52, 55]. Ein Hauptnachteil ist darin zu sehen, dass die Flexibilität solcher Systeme durch die erforderliche Neukalibrierung bei Veränderung der Zellengeometrie oder von Kameraparametern wesentlich eingeschränkt ist. Um diesen Nachteil der konventionellen Methode zu beseitigen und die Flexibilität der Industrieroboter zu erhöhen, können neuronale Netze als ein Werkzeug zum Lernen verwendet werden. Eine DARPA- Studie [57] zeigt, dass die Robotik, besonders um die Flexibilität der Roboter zu erhöhen, breite Anwendungsmöglichkeiten für den Einsatz neuronaler Netze bietet. Über den gegenwärtigen Entwicklungsstand und weitere Entwicklungstendenzen dazu gibt Huang in [41] einen guten Überblick. Dabei werden zum gegenwärtigen Zeitpunkt existierende Anwendungen neuronaler Netze und mögliche weitere Verbesserung in der Industrie dargestellt.
Zalzala [102] liefert einen Überblick zu den derzeit benutzten systemtheoretischen Konzepten und deren Anwendung beim Einsatz neuronaler Netze in der Robotik. Zur Verbesserung der Flexibilität visueller Steuerung sind lern- und adaptionsfähige Systeme nach [83] erwünscht. In diesem Gebiet existieren einige interessante Untersuchungen, die auf Basis neuronaler Methode durchgeführt wurden. Miller [61, 62] verwendet eine Konfiguration mit einer auf der Roboterhand aufgesetzten Kamera zur Verfolgung eines auf einem Förderband befindlichen Objektes. Dabei soll der Roboter so angesteuert werden, dass sein Endeffektor eine entsprechende relative Position und Orientierung zum Objekt erreicht. Dazu wird eine Zuordnung von Abweichung zwischen dem Ist- und Sollbild zu den Gelenkkoordinaten des Roboters von einem CMAC- Netz gelernt.

In [38] wird ein Konzept zur Ansteuerung des Roboters mittels visueller Information beschrieben. Die nichtlineare Relation zwischen den Bilddaten und der Veränderung der Gelenkwinkel wird durch ein Backpropagation (BP)-Netz gelernt. Die Eingangsinformationen für das BP-Netz werden von einer am Endeffektor aufgesetzte Kamera geliefert. Kuperstein [51] verwendet ein nichtlineares Netz zur Bestimmung einer Karte, die 3D-Koordinaten direkt den entsprechenden Gelenkwinkeln zuordnet. Damit kann der Roboter zum Greifen einfacher, z.B. zylindrischer Körper im Raum beliebig positioniert werden. Es werden zwei festpositionierte Kameras verwendet. Zur visuellen Steuerung von Positionierbewegungen erlernt das Netzwerkmodell durch entsprechende Trainingsbewegungen die Kinematik des Arms und die optischen Abbildungseigenschaften der festpositionierten Kameras. In [94] erfolgen dazu experimentelle Untersuchungen, wobei deutlich wird, dass die Genauigkeit der Positionierung durch die verwendeten festpositionierten Kameras begrenzt ist. Es wurde von Van der Smagt [85] und von Sun [88] theoretisch demonstriert, dass ein visuell gesteuerter Roboter mit Hilfe eines neuronalen Netzes positionieren kann.
Die angeführten Quellen zeigen, dass lernfähige Strukturen in Form neuronaler Netze ein besonders interessantes Werkzeug zur Erhöhung der Flexibilität und Autonomie von Robotern sind. Zur Zeit sind solche Strukturen nur für einen kleinen Beobachtungsraum der Kameras bzw. für eine kleine Anzahl der Gelenke realisierbar, da die Datenmengen (verschiedene Positionen und Orientierungen des Objektes) und der Rechenaufwand (beim Training des Netzes) sonst zu groß werden. Dabei ist die Positionierungsgenauigkeit ziemlich klein und für die praktischen Anwendungen in der Industrie kaum geeignet. Deshalb wird in dieser Arbeit ein Konzept der visuellen Steuerung vorgeschlagen (siehe Abschnitt 2.3), der im Unterschied zu neuronalen Netzen auch mit moderner Hardware in Echtzeit zu realisieren ist.
Bei der visuellen Steuerung kann man zwei Konfigurationen, einmal mit festpositionierten Kameras und zum anderen mit an der Roboterhand aufgesetzten Kameras, unterscheiden. Während festpositionierte Kameras den Nachteil einer zu geringen Genauigkeit der Objektbewegung haben, ist bei aufgesetzten Kameras der Erfassungsbereich eingeengt und somit ihr Arbeitsbereich stark begrenzt. Für viele Anwendungen ist es erforderlich, dass der Roboter sich aufgrund visueller Information in einem großen Arbeitsraum mit erforderlicher Genauigkeit bewegen kann. Dies kann durch eine Kombination beider Kamerakonfigurationen realisiert werden. Aus diesem Grund wird diese Kombination für das zu entwickelnde visuelle Robotersteuerungskonzept vorgeschlagen.

Visual Servoing:

Die meisten bisherigen Arbeiten haben nur die klassischen Aufgaben des *Visual Servoing* untersucht. Dies betrifft die kinematische Relation zwischen Objektkonfiguration, Roboterkonfiguration und visuellen Merkmalen ohne Berücksichtigung der dynamischen Eigenschaften. Die *Visual Servoing* Methode, die rein auf kinematischer Betrachtung basiert, ist jedoch nur für die Anwendungen mit niedriger dynamischer Anforderung ausreichend. Zur Verbesserung der Eigenschaften des *Visual Servoing* ist die Dynamik des Visual Servoing-Systems im Regelungskonzept zu berücksichtigen [20, 42].
Zwei verschiedene Verfahren, und zwar ein positionsbasierendes [97] und ein bildbasierendes Verfahren [24, 95], werden in der Regel im *Visual Servoing* verwendet. Im bildbasierenden Verfahren können die visuellen Merkmale durch die Eckpunkte oder die Flächen ermittelt werden. Die Elemente der visuellen Merkmale sind daher nicht im

Roboterraum sondern im Bildraum definiert. Nach der Aussage von Skaar [84] können viele im Roboterraum definierte Aufgaben durch eine oder mehrere Aufgaben im Bildraum dargestellt werden. Im positionsbasierenden Verfahren werden dagegen die visuellen Merkmale im Roboterraum definiert.
Ende der siebziger Jahre beschrieb Rosen in [75, 76] die Verwendung visueller Rückführung für die Einfügung eines Bolzens und die Aufnahme von auf einem Förderband liegenden Teilen. Hill und Park [39] stellten 1979 *visual servoing* für einen Unimate- Roboter dar. Dabei wurde die Binärbildverarbeitung zur Ermittlung planarer Positionen sowie zur Abschätzung der Tiefe durchgeführt. Weiss [95] stellte eine bildbasierende *Visual Servoing* Struktur vor, in der Achswinkel des Roboters durch die erfassten visuellen Merkmale ermittelt werden. Dabei beinhaltet die berücksichtigte Nichtlinearität die Roboterkinematik und -dynamik sowie das Bildmodell. In der Simulation ist das vorgestellte Konzept für verschiedene Manipulatorstrukturen mit bis zu 3 Freiheitsgraden untersucht. Die Arbeit von Weiss wurde von Feddema [27] in verschiedener Hinsicht erweitert. Dies wurde besonders durch Experimente demonstriert. In der Arbeit wurde gezeigt, dass die Abtastzeit des visuellen Systems wesentlich größer als die Abtastzeit der Roboterregler ist. Dies führt zu relativ langsamen Geschwindigkeiten und zur schlechten Genauigkeit der Objektverfolgung. Jang [44, 45] führte das Konzept des vergrößerten Bildraums und des transformierten Merkmalsraums ein. Der vergrößerte Bildraum ist ein 3D-Raum, dessen Koordinaten die Bildebenekoordinaten und die Abstände zur Kamera sind, die mit Hilfe von so genanntem Bewegungsstereo bestimmt werden.
Visual Servoing ist zur Zeit ein aktuelles Forschungsfeld [24, 38, 46, 47, 53, 63, 86]. Dafür wurden verschiedene Methoden entwickelt und untersucht. In [86] wurden neuronale Netze zur Merkmalsextraktion und Annäherung der inversen Jacobimatrix für die bildbasierende visuelle Steuerung verwendet. Ein prädikatives Schätzverfahren wurde von Piepmeier [73] für die Verfolgung eines bewegten Objektes vorgeschlagen. Eine auf einem Störungsbeobachter basierende Methode wurde von Lee [53] entwickelt.
Aufgrund der hohen dynamischen Anforderungen von zukünftigen Anwendungen ist es sinnvoll, die dynamischen Eigenschaften eines visuell gesteuerten Robotersystems zu berücksichtigen.
Die meisten konventionellen visuellen Systeme sind nicht schnell genug für *Visual Servoing* des Roboters, weil die Feedbackabtastung durch die Bildabtastzeit und die Bildverarbeitungsgeschwindigkeit begrenzt ist. Zur Zeit werden immer schnellere Bildverarbeitungssysteme mittels neuer Rechnertechnik entwickelt und hergestellt [21, 25, 48, 98], die aber sehr teuer sind. Dafür wurde ein massiv paralleles Bildverarbeitungssystem von Nakabo [64] entwickelt. Darin sind die Photo-Detektoren und die Verarbeitungselemente direkt verbunden. Die Bildelemente sind in einem Chip integriert, so dass der Flaschenhals der Bildübertragung nicht mehr existiert. Daher ist es möglich, ein schnelles (1ms) visuelles Steuerungssystem zu realisieren. Dies bietet eine Realisierungsmöglichkeit, mit der schnelle visuelle Regelungskonzepte für die zukünftigen Anwendungen entwickelt werden können.

2.2 Grundgedanke des Konzepts

Die Robotersteuerung mittels visueller Information erfordert den Aufbau einer Verbindung zwischen visueller Information und den Achswinkeln bzw. zwischen visueller Information

und den Achsmomenten. Zur Erfassung dieser Verhältnisse spielen das kinematische und das dynamische Verhalten des Roboters eine wesentliche Rolle (siehe Abb. 2.1).

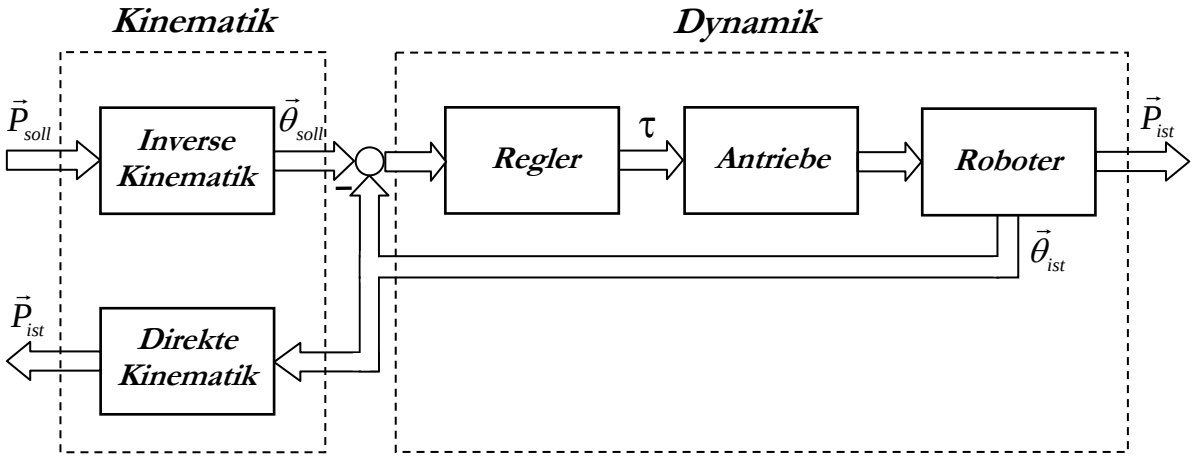


Abbildung 2.1: Das kinematische und dynamische Modell

Das kinematische Modell beschreibt die Relation zwischen den kartesischen Raumkoordinaten und den entsprechenden Gelenkkoordinaten des Roboters. Der Zusammenhang zwischen den Gelenkkoordinaten sowie deren Ableitungen und den Achsmomenten wird durch das dynamische Modell dargestellt. Um eine Robotersteuerung zu realisieren, muss die Betrachtung vorangestellt werden, dass, unabhängig vom Typ eines mechanischen Manipulators, die Aufgabenspezifikation grundsätzlich im durch die kartesischen Koordinaten des Raumes $\vec{P}$ definierten *Arbeitsraum* des Roboters vollzogen wird, während die eigentliche Regelaktion (die Kräfte bzw. Momente der Gelenkaktuatoren) im Raum der Gelenkwinkel $\vec{\theta}$, dem so genannten *Konfigurationsraum* des Roboters erfolgt. Zunächst wird die inverse Kinematik des Manipulators gelöst, um die geforderten Bewegungen $\vec{P}_{soll}$ aus dem Arbeitsraum in Variablen $\vec{\theta}_{soll}$ des Konfigurationsraums zu transformieren. Diese werden durch die eigentliche Roboterregelung umgesetzt. Dabei erfolgt die Bestimmung der durchgeführten Arbeitsraumvariablen $\vec{P}_{ist}$ in der Regel nicht direkt, sondern mittels einer Berechnung der direkten Kinematik aus den gemessenen Gelenkvariablen $\vec{\theta}_{ist}$.

Für die in der Literatur beschriebenen Verfahren zur visuellen Steuerung sind die Bildverarbeitung, die aufwendigen Kalibrierungen der Kameras und kinematische Transformationen erforderlich. Dabei sind auch zum einen die Roboter- und Kameraparameter besonders schwierig präzise zu ermitteln. Zum anderen können sich die Bedingungen des Roboters oder der Kamera durch Verschleiß bzw. externe Störungen geändert haben. In solchen Fällen muss das ermittelte Modell neu gebildet werden.

Die derzeit verfügbaren Bildverarbeitungssysteme sind aufgrund der niedrigen Bildabtastzeit und der erforderlichen Verarbeitungszeit im Vergleich zur Roboterdynamik langsam, was zur Verlangsamung der Geschwindigkeiten bei der Objektverfolgung führt. Deswegen kann oftmals die Dynamik des Roboters vernachlässigt werden. Für die praktische Untersuchung muss diese Tatsache beim Entwurf des optisch geführten Steuerungskonzeptes berücksichtigt werden.

Durch die Beobachtung der menschlichen Hand-Augen-Koordination ist allgemein bekannt, dass ein Bearbeitungsvorgang in der Regel aus drei Untervorgängen besteht, nämlich dem

schnellen Primärbewegungsvorgang zur groben Platzierung, dem langsamen Justiervorgang für die feine Justierung und dem Operationsvorgang zum Behandeln von Gegenständen. Für bestimmte Aufgaben könnte der eine oder andere Vorgang ausfallen. Beispielsweise ist der Justiervorgang nicht mehr notwendig, wenn die Aufgabe der Objektverfolgung keine hohe Genauigkeitsanforderung verlangt. In Abb. 2.2 ist die Funktionsstruktur solcher Steuerung dargestellt. Dabei besteht der Bearbeitungsvorgang ebenfalls wie bei Menschen aus drei Untervorgängen, und zwar einem Primärvorgang zur Ermittlung der groben Position und Orientierung des Roboters, einem Justiervorgang und einem Operationsvorgang zur Ansteuerung des entsprechenden Werkzeugs.

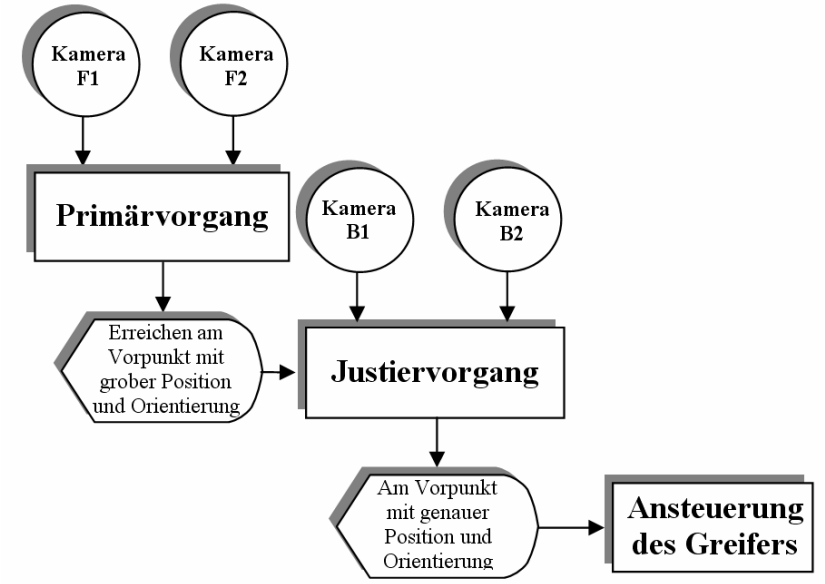


Abbildung 2.2: Funktionsstruktur der visuellen Steuerung

Die Erkennung und die Lagebestimmung von starren, dreidimensionalen Objekten, die in ihrer geometrischen Form meistens vollständig bekannt sind, stellt insbesondere im Bereich der Automatisierungstechnik ein in vielen Varianten auftretendes Problem dar. Trotz seiner großen praktischen Bedeutung kann dieses Problem jedoch bis heute nur in recht spezialisierten Anwendungsfällen als gelöst betrachtet werden. Ein näherer Blick in die Recht umfangreiche Literatur zu dieser Thematik macht dabei deutlich, dass die Leistungsfähigkeit derartiger Erkennungsverfahren zwar zum einen von den jeweils verwendeten Erkennungsalgorithmen mitbestimmt wird, in einem weitaus größeren Maße jedoch von der Art, Dichte und Genauigkeit der zugrunde liegenden Sensorinformation abhängt.
Die 3D- Bildverarbeitung erfordert mindestens zwei Aufnahmebilder von verschiedenen Ansichten. Diese können durch eine Kamera zweimalig von unterschiedlichen Standpunkten oder durch zwei Kameras einmalig erzeugt werden. In der visuellen Steuerung von Robotern muss die relative Lage zwischen dem Endeffektor und den Kameras relativ beibehalten werden, so dass die von den Kameras erfasste Information durch die Transformation direkt in der Robotersteuerung genutzt werden kann. Daher ist es erforderlich, mindestens zwei Kameras zur Ermittlung der 3D-Raumkonfiguration in der visuellen Steuerung zu verwenden.
Um den Gesamtarbeitsraum beobachten zu können, werden zunächst zwei am Rand des Arbeitsraums festpositionierte Kameras *F1* und *F2* verwendet. Wegen der niedrigen Kameraauflösung und der großen Abstände zwischen Zielkonfigurationen und den

feststehenden Kameras sowie weiterer Störungen ist die Messgenauigkeit dieser beiden fixierten Kameras begrenzt. Aufgrund dessen ist es darüber hinaus notwendig, für die Aufgaben, die hohe Genauigkeit erfordern, zwei zusätzliche am Roboterendeffektor montierte Kameras *B1* und *B2* einzusetzen. Basierend auf der visuellen Information dient der Grundvorgang zur Ermittlung der groben Position und Orientierung des Endeffektors. Dabei werden die visuellen Informationen aus den zwei zum Arbeitsraum festpositionierten Kameras zur Erfassung des Gesamtarbeitsraums genutzt. In diesem Prozess wird der Roboter so gesteuert, dass das Zielobjekt von den (am Endeffektor aufgesetzten) beweglichen Kameras erfasst werden kann. Als Ausgabewerte muss der Grundvorgang mittels der durch Bildkoordinaten ausgedrückten Objektposition und –orientierung demnach die passenden Gelenkkoordinaten zum Aufsuchen dieser Lage liefern. Ähnlich den optischen Trackern bieten sich für die beweglichen Kameras sowohl die Outside-in Variante als auch die Inside-out Variante an. In unserem Fall werden Kameras am Objekt befestigt, um während der wissenschaftlichen Untersuchungen die Kollisionen der Gelenke des Roboters mit den Kameras zu vermeiden. Auch die Montage der Kameras auf dem Endeffektor sind problematisch.
Nach einer abgeschlossenen groben Platzierung vom Roboter wird der Prozess zur genaueren Justierung gestartet. Dabei wird durch diesen Justiervorgang eine weitere Gelenkwinkeländerung aufgrund der relativen visuellen Abweichung geliefert. Mittels dieses Justiervorganges lässt sich die Lage des Endeffektors durch Steuerung der Gelenkkoordinatenänderung auf Basis visueller Informationen aus den am Objekt aufgesetzten zwei Kameras so fein justieren, dass der Endeffektor des Roboters mit hinreichender Genauigkeit in die für die Operation erforderliche Position und Orientierung platziert wird. Dabei muss der Roboter einen relativen Abstand des Zielobjektes bzgl. des Endeffektors ansteuern und diesen Abstand bei Bewegung des Objektes halten. In dieser Arbeit werden der Primärvorgang und der sich anschließende Greifvorgang nicht in Betracht gezogen, sondern es wird angenommen, dass sich der Endeffektor bereits im Sichtbereich der auf dem Objekt aufgesetzten Kameras befindet und durch diesen Justiervorgang so positioniert und orientiert wird, dass ein Greifvorgang (auch bei Objektverfolgung) ausgeführt werden kann. Im nächsten Abschnitt wird eine solche Struktur der optisch geführten Steuerung entwickelt und ausführlich beschrieben.

2.3 Struktur der optisch geführten Robotersteuerung

Die Robotersteuerung unter Nutzung optischen Detektoren erfordert den Aufbau eines Systems zur Kopplung visueller Informationen mit den Gelenkwinkeln. Die Abbildung 2.3 stellt die vorgeschlagene Struktur eines Systems dar, die aus dem Kameramodell, der direkten Kinematik, dem Regler, dem Trajektoriengenerator 1 und 2 und der inversen Kinematik besteht. Dieses Signalflussbild der Steuerung wurde für zwei am Objekt festpositionierten Kameras zur präzisen Bestimmung der Position und der Orientierung des Objektes entwickelt. Ein ähnliches Konzept kann aber auch auf das Kamerasystem mit zwei zusätzlichen am Rand des Arbeitsraumes des Roboters befestigten Kameras für die äußere Beobachtung übertragen werden.
Es wird in dieser Arbeit davon ausgegangen, dass mit Hilfe eines Primärvorganges die Position und Orientierung des Zielobjektes $\vec{P}_o^*$ schon grob bekannt sind. Diese Werte neben der Eingabe des gewünschten relativen Abstands zwischen dem Objekt und dem

Endeffektor $\Delta\vec{P}_{soll}$ dienen für die *Trajektoriengenerierung 1*. Der Block *Trajektoriengenerierung 1* erzeugt eine räumliche Trajektorie $\vec{P}_{soll}$, die den Endeffektor mit einem gewünschten Abstand vom Objekt positionieren wird.

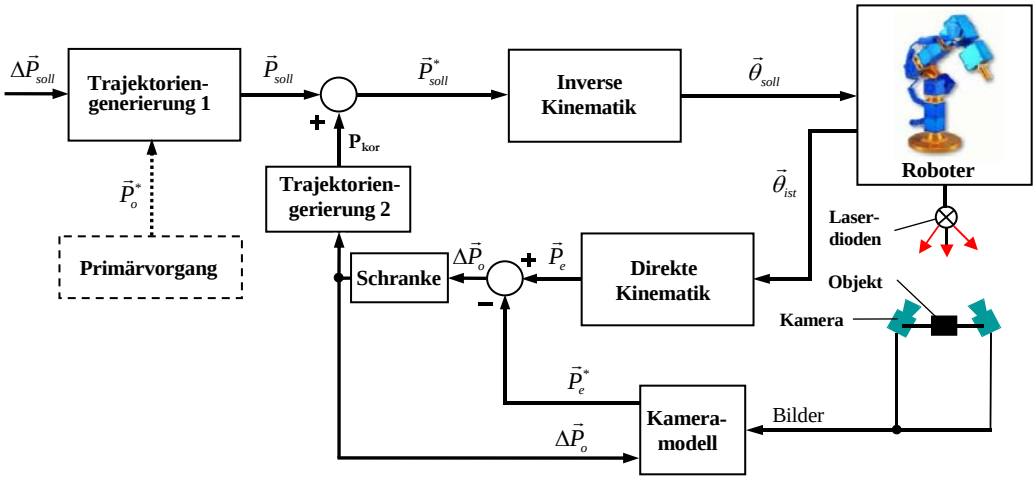


Abbildung 2.3: Struktur der optisch geführten Robotersteuerung

Da die Trajektorien im kartesischen Arbeitsraum des Roboters generiert werden, muss der Verlauf der Sollposition und –orientierung des Endeffektors mittels der *inversen Kinematik* in die erforderlichen Gelenkwinkel des Roboters $\vec{\theta}_{soll}$ transformiert werden. Diese werden durch die eigentliche Roboterregelung umgesetzt, damit wird letztendlich der Endeffektor in der gewünschte Lage positioniert. Dabei erfolgt die Bestimmung der durchgeführten Arbeitsraumvariablen des Endeffektors $\vec{P}_e$ in der Regel nicht direkt, sondern mittels einer Berechnung der *direkten Kinematik* aus den mittels der Inkrementalgeber gemessenen Gelenkvariablen $\vec{\theta}_{ist}$. Anderseits werden die Position und Orientierung des Endeffektors $\vec{P}^*_e$ mittels eines *Kameramodells* ermittelt, das dazu die Signale der optischen Detektoren verwendet. Wenn sich das Objekt bewegt oder wegen anderer Ursachen eine ungenaue Positionierung erfolgt, ist die Differenz $\Delta\vec{P}_o$ zwischen den realen $\vec{P}_e$ und visuellen $\vec{P}^*_e$ Werten der Lage des Endeffektors im Arbeitsraum verschieden von Null. Dabei sind die mittels visueller Information ermittelte Position und Orientierung des Endeffektors mit den Messstörungen der Kameras (siehe Abschnitt 6.2) behaftet, was sich negativ auf die Genauigkeit auswirkt. Deshalb wird das Differenzsignal $\Delta\vec{P}_o$ mit Hilfe einer Schranke von diesen Störungen befreit. Die Größe der Schranke muss vorerst experimentell bestimmt werden, was im Kapitel 7 ausführlich beschrieben wird. Dieses Differenzsignal bildet die Veränderung der Lage (Bewegung) des Objektes in einem Takt der Steuerung ab, und dient als Eingangssignal für den *Trajektoriengenerator 2*. Da dieses Differenzsignal sich sprunghaft ändert, da das Kameramodell viel langsamer (siehe Kapitel 6) als der Rest der Elemente der optisch geführten Steuerung funktioniert, wird der Trajektoriengenerator 2 eine sanfte Korrekturtrajektorie $\vec{P}_{kor}$ erzeugen. Die bei Addition neu entstehende Sollposition und –orientierung $\vec{P}^*_{soll}$ bezieht die Korrekturbahn des bewegten Objektes ein.

Zur Verdeutlichung der Wirkungsweise der optisch geführten Robotersteuerung, stellt Abbildung 2.4 eine vereinfachte Struktur solch einer Steuerung dar. Wie schon erwähnt

wurde, lässt diese Struktur sowohl die visuelle Information über die Lage des Objektes bzgl. der Kameras als auch die mittels der direkten Kinematik errechnete Position und Orientierung des Endeffektors bei Bewegung des Objektes zu. Wenn sich das Objekt nicht bewegt, wird das Differenzsignal $\Delta\vec{P}_o$ jedoch wegen der Messstörungen der Kameras nicht nahezu Null sein. Mit Einbau einer Schranke werden diese Störungen unterdrückt, dabei verschwinden die Auswirkungen der Rückführung in der optisch geführten Robotersteuerung. In diesem Fall kann man dann von einer offenen Robotersteuerung sprechen.

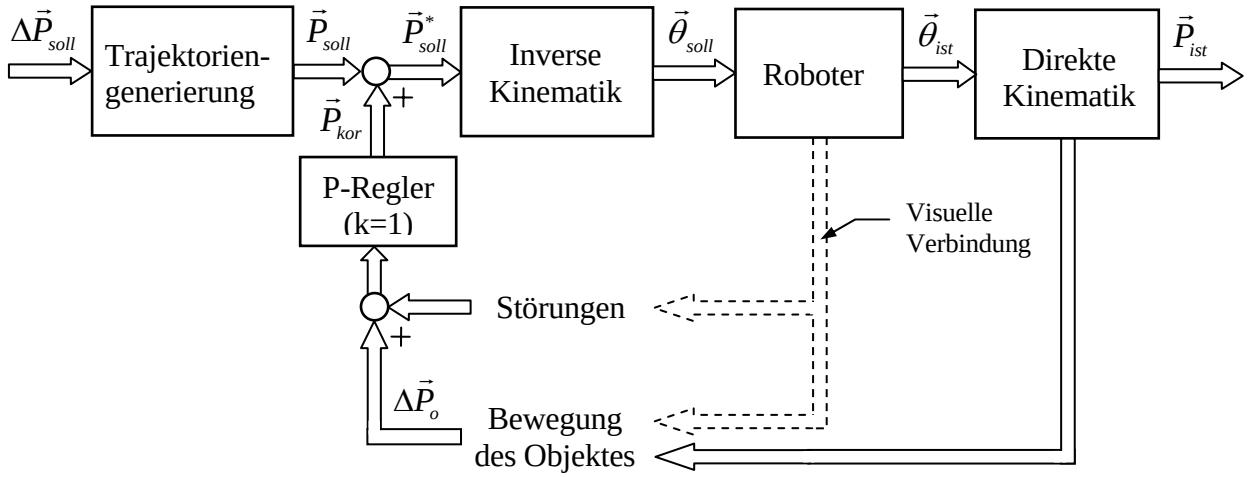


Abbildung 2.4: Vereinfachte Struktur der optisch geführten Robotersteuerung

Bei der Bewegung des Objektes wird die Rückführung jedoch wieder wirken. In diesem Falle kann man also nicht mehr von einer offenen Robotersteuerung, sondern muss von einer optisch geführten Roboterregelung sprechen. Wegen der komplizierten Nichtlinearität dieser Steuerung vor allem in der inversen Kinematik, der Dynamik des Roboters und des Kameramodells wird von einem herkömmlichen Entwurf eines nichtlinearen Reglers abgesehen. Es wird in dieser Arbeit auch keine Robustheits- und Stabilitätsanalyse der gesamten Struktur der optisch geführten Robotersteuerung durchgeführt. Für die weiteren Untersuchungen wurde ein P- Regler mit *k=1* festgelegt. Damit wird in dieser Arbeit im Weiteren von einer übergeordneten Robotersteuerung gesprochen.
Im Kapitel 7 werden die experimentellen Untersuchungen dieser Struktur der visuellen Robotersteuerung bei verschiedenen Geschwindigkeiten des Objektes durchgeführt, um experimentell die Robustheit und Stabilität eines solchen Systems nachzuweisen und die maximale Geschwindigkeit der Objektverfolgung zu bestimmen.

Kapitel 3

Hardwarerealisierung des Systems zur praktischen Verifizierung

Die Anforderungen an die Geräte und an ihren Entwurf ändern sich im Vergleich zu Industrierobotern. Anstatt, wie dies bisher Stand der Technik ist, die Umgebung an den Roboter anzupassen, ist es bei vielen Aufgaben notwendig, spezielle, an die Umgebung angepasste, Roboter zu entwerfen. Diese Vorgehensweise ist notwendig, weil die Geräte in einer vorgegebenen Umgebung eingesetzt werden sollen, die nicht beliebig verändert werden darf: Sowohl die Akzeptanz der Roboter als auch der Grad, in dem sie die Aufgabe erfüllen können, hängt stark davon ab, inwieweit der Roboter in die Umgebung integriert werden kann. Die Integration wiederum ist nur möglich, wenn sowohl die Schnittstelle zur Bedienung des Roboters als auch sein Aufbau an die Aufgabe angepasst werden kann. Genau deshalb steht der Hardwareentwurf soweit im Mittelpunkt dieser Arbeit.
Kinematisch redundante Manipulatoren, das heißt Manipulatoren, bei denen die Anzahl der Freiheitsgrade (Gelenke) n größer ist als die Dimension des Aufgabenraumes, sind in besonderem Maße für diese Art der Anpassung geeignet. Sie sind in der Lage, Aufgaben auszuführen, die nichtredundante Roboter nicht bewältigen können. Der Aufgabenraum kann zum Beispiel der dreidimensionale kartesische Raum sein. In diesem, im weiteren vorausgesetzten Fall hat er die Dimension $m=6$. Weitere $n-m$ Freiheitsgrade erlauben es dann, zum Beispiel Hindernissen auszuweichen, singuläre Konfigurationen zu vermeiden, trotz Gelenkbeschränkungen einen genügend großen Arbeitsraum zu erreichen oder die Beweglichkeit des Roboters zu optimieren. Damit ist es möglich, auf zusätzliche Sensorinformationen reagieren zu können. Diesen Vorteilen stehen allerdings Nachteile gegenüber: Redundante Manipulatoren sind sehr viel aufwendiger zu entwerfen und zu steuern als nichtredundante Roboter [66], [74].
Die Entwicklung der Antriebskomponenten in den letzten Jahren verdeutlicht die wachsende Bedeutung von Servo-Antriebssystemen in der Robotertechnik. Die Vorteile gegenüber konventionellen elektrischen Antrieben liegen auf der Hand:

- Servomotoren verfügen über eine sehr hohe Dynamik
- Servoantriebe gestatten den Aufbau einer präzisen Positionsregelung ohne Verluste in der Dynamik.
- Servoantriebe mit bürstenlos kommutierten Motoren haben eine hohe Lebensdauer
- Servoantriebe werden in immer kleineren Abmessungen und hoher Leistung verfügbar.

Mit dem Einsatz der Servoantriebe wächst der Trend zur Verlagerung von immer mehr Intelligenz in die Antriebssteuerung, die mithin zu einem Teil der Robotersteuerung wird.

Gerade bei herkömmlichen Positionssteuerungen ergeben sich bislang Schwierigkeiten in der Anwendung der modularen Bauweise, weil die Größe der Regler- und Leistungselektronik, die Größe des Motors oft weit überschreitet. Dies hat die Folge, dass die Positionsregler in Schranksystemen eingebaut und aufwendig über weite Distanzen mit dem Motor verkabelt werden. Insbesondere bei Robotersteuerungen mit mehreren Achsen ist es wünschenswert, die intelligenten Module miteinander zu vernetzten und an die Robotersteuerung zu koppeln.
Aus diesen Gründen wurde ein redundanter (7-achsiger) wissenschaftlicher Roboter *amtec r7* aufgebaut, der dem Zwecke der Entwicklung einer optisch geführten Steuerung dienen soll. Zentrales Grundelement des Roboters *amtec r7* sind kompakte Antriebsmodule (siehe Abb. 3.1), die bei minimalem Platzbedarf in Doppelwürfeln leistungsfähige Servomotor-Getriebe-Kombinationen, die zugehörige Leistungselektronik, verschiedene Sensoren (Winkelencoder, Stromwandler, Temperaturfühler), einen Mikrocontroller zur Bewegungssteuerung und Zustandsüberwachung sowie ein intelligentes Businterface (CAN) integrieren. Mit diesen Modulen und verschiedenen mechanischen Verbindungs- und Adapterelementen können vielfältige kinematische Strukturen aufgebaut werden (siehe Abb. 3.2). Die elektrische Verbindung für Energieversorgung und Kommunikation der Module erfolgt über einheitliche Kabel mit Steckverbindern entlang der kinematischen Kette der Roboterstruktur.

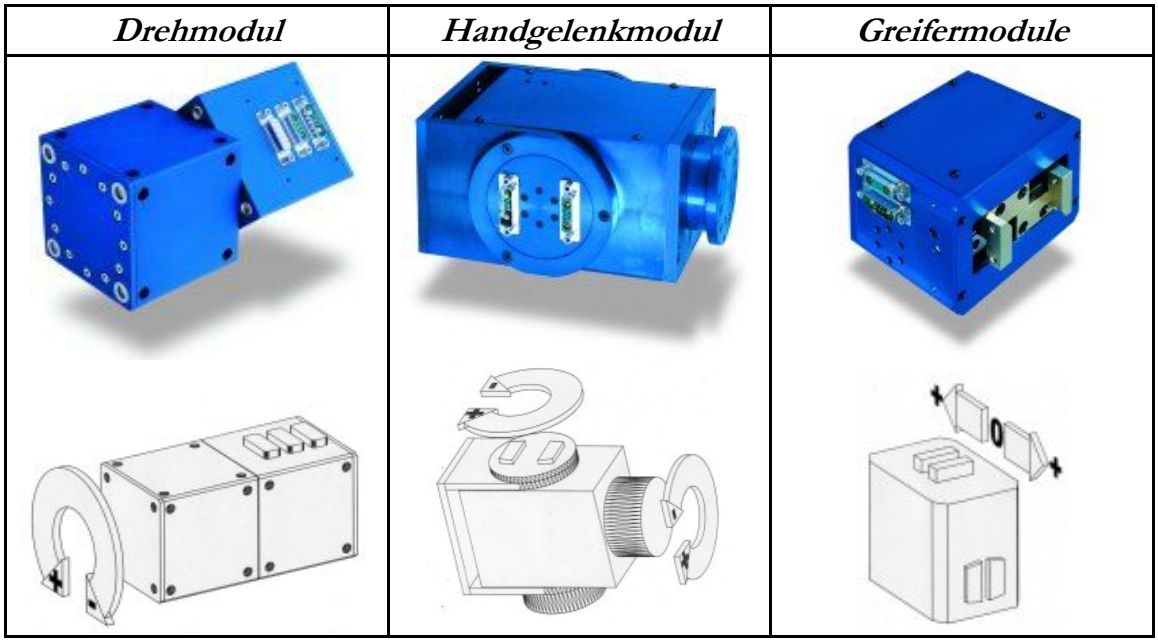


Abbildung 3.1: Die Robotermodule

Der Vorteil des aufgebauten Roboters *amtec r7* besteht in der Möglichkeit der dezentralen Steuerung über einen CAN-Bus oder eine RS232-Komunikations-Schnittstelle. Zu seinen Besonderheiten gehört auch, dass alle Elemente der Steuerung im Roboter integriert sind, dies bezieht sich auf die Inkrementalgeber zur Bestimmung der Gelenkwinkel sowie den volldigitalen Positions-, Geschwindigkeits- und Stromregler. Ein weiterer Vorteil dieses Roboters ist, dass der Regelkreis der Motoren mit einer Tastzeit von 0.25ms arbeitet. Dabei liegt die Positionsgenauigkeit des Roboters bei ±0.1mm. Über den CAN-Bus können die Soll- und Istgelenkwinkeln in einem 1ms Zyklus zwischen dem Rechner und dem Roboter

ausgetauscht werden. Alle diese Vorteile sind besonders wichtig für die Realisierung der optisch geführten Robotersteuerung.

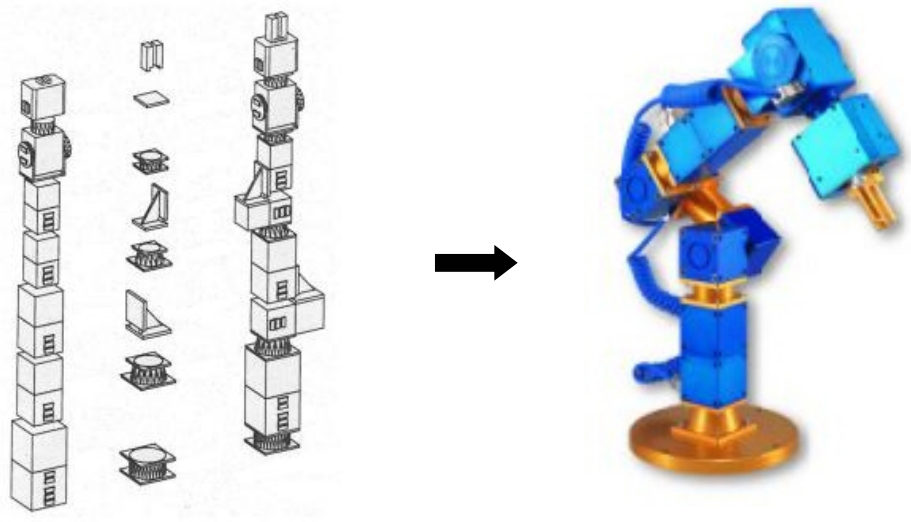


Abbildung 3.2: Aufbau eines Roboters aus den Modulen

Zur Ermittlung der Position und Orientierung des Objektes mittels visueller Information in der bereits erläuterten *Outside-in* Anordnung (siehe Kap. 2) wurden 4 Laserdioden auf dem Endeffektor befestigt, um einen genügend großen Beobachtungsraum der Kameras zu ermöglichen, die notwendige Abtastzeit des Kamerasystems sicher zu stellen und eine näherungsweise Unabhängigkeit von weiteren äußeren Lichtquellen zu erreichen. Für den Einsatz wurde der kleine Öffnungswinkel der Laserdioden durch Streulinsen vergrößert. Um die Laserdioden-Wirkung mit den Kameras zu erfassen, wurden Modulationssignale für die Laserdioden mit Hilfe eines DSpace- Signalprozessorsystems erzeugt. Diese Signale schalten nacheinander die Laserdioden mit einer Periode von 20ms ein und aus. Die relativ niedrige Schaltfrequenz der Laserdioden ist damit verbunden, dass die Dioden bei deren Modulation nicht schneller ein- und ausgeschaltet werden können. Anderenfalls würde das Restlicht der Dioden sehr stark die Signale der anderen Dioden beeinflussen, was zur Störungen der Kamerasignale führt. Da die Laserdioden die notwendige Information zur Berechnung der Position und Orientierung des Endeffektors liefern, werden die berechneten stark von realen Werten unterscheiden.

Um die Position eines Lichtpunktes auf einer optischen Detektoroberfläche zu bestimmen, gibt es prinzipiell zwei Möglichkeiten: Entweder betrachtet man die gesamte Fläche und versucht festzustellen, was darauf ein Lichtpunkt ist und wo er sich befindet. Oder aber man verwendet eine Vorrichtung, die allein auf eine Lichtpunkt-Position reagiert und alle andere Größen ignoriert.
Die erste Methode entspricht der Arbeitsweise eines CCD (charge coupled device – Zellen- oder Bildsensor mit Ladungsverschiebung), die zweite die eines PSD (position sensitive detector – positionsempfindlicher Sensor).
PSDs und CCDs sind zwei grundsätzlich verschiedene Bauelemente. Zwar detektieren beide die Position von einfallendem Licht aber in sehr unterschiedlicher Weise. Das Ausgangssignal eines PSD hängt von der Position des Schwerpunktes der gesamten Lichtverteilung auf der aktiven Fläche ab. Ein CCD dagegen erfasst das Licht in vielen getrennten Pixeln parallel und gibt dann die einzelnen Messwerte sequentiell aus. Daraus kann ein Rechner dann den Maximalwert und dessen Position ermitteln.

PSDs arbeiten örtlich und zunächst zeitkontinuierlich analog. Sie werten den Strom aus, der von einer Fotodiode erzeugt wird und sich in eine oder zwei resistive Schichten aufteilt. Die Vorteile dieser einfachen Bauweise sind Stabilität und Zuverlässigkeit. Die für die Verarbeitung des analogen Ausgangssignals benötigte Elektronik ist relativ einfach und lässt sich vergleichsweise kostengünstig aufbauen.
Ein CCD ist im Prinzip ein Feld aus sehr vielen dicht nebeneinander liegenden einzelnen MOS- Dioden. Das Licht erzeugt in jeder Diode eine elektrische Ladung. Durch Anlegen einer geeigneten Folge von Taktimpulsen werden die angesammelten Ladungen über ein Chip hinweg zu dessen Ausgang übertragen, wo sie in Spannungen gewandelt werden. Die komplizierte Struktur macht CCDs schwieriger zu fertigen und anfälliger gegenüber Defekten. Sie geben ein örtlich bzw. zeitlich diskretes Ausgangssignal ab.
Ein PSD ermittelt allein die Position des Schwerpunktes des auftretenden Lichtpunktes. Das tut er allerdings innerhalb von Nanosekunden und mit Sub-Nanometer-Auflösung. Er erreicht dabei eine Messtoleranz von etwa 0,1% der Sensorgröße.
Da der PSD die Positionsinformation aus den Photoströmen der Dioden gewinnt, lassen sich hier Betriebsweisen wie bei normalen Photodioden anwenden– etwa eine hochfrequente Modulation des Lichtes, um Störungen durch Fremdlicht zu eliminieren.
Das Ausgangssignal eines CCD enthält Informationen über die Lichtintensitätsverteilung auf der gesamten aktiven Fläche, es beschreibt also ein Bild. Ein CCD ist deshalb die übliche Wahl für den Bildsensor in einer Videokamera. Den Schwerpunkt eines Lichtpunktes kann ein CCD aber nicht ohne zusätzliche digitale Signalverarbeitung ermitteln. Damit ist dieser Messwert nicht so schell verfügbar wie beim PSD. Alle Pixel abzutasten und digital zu verarbeiten, benötigt einige Zeit und macht das CCD sehr viel langsamer als den PSD. Anderseits haben die Pixel eine durch die Herstellungsmaske definierte Position, daraus resultiert eine sehr hohe Genauigkeit. Um jedoch die maximale Genauigkeit und die höchste Auflösung zu erhalten, ist eine Interpolation zwischen benachbarten Punkten erforderlich. Dies verlangsamt diese Messung noch weiter. Für Lichtpunkte, die kleiner sind als der Abstand zwischen zwei benachbarten Pixels, ist keine Interpolation möglich, und das Signal geht verloren. Dies setzt eine untere Grenze für die Größe des verwendbaren Lichtpunktes.
Der Dynamikbereich eines CCD ist begrenzt, und eine plötzliche Änderung der Lichtintensität kann ein „Blooming“ verursachen – eine Überstrahlung benachbarten Pixels. Ein Vorteil des CCD ist (wie beim menschlichen Auge), dass es empfangenes Licht bis zum Zeitpunkt der Messung speichern kann. Diese Eigenschaft ist bei extrem kleinen Lichtintensitäten von Nutzen.
Entsprechend dem oben beschriebenen Vergleich wurden im Sinne der visuellen Robotersteuerung die positionsempfindlichen Detektoren zur Messung der Lichtsignale ausgewählt und weiter verwendet.

Ein (PSD) arbeitet ähnlich wie eine normale Photodiode. Das auf das aktive Gebiet fallende Licht generiert einen Photostrom, der in Richtung des p- und des n-Gebietes abfließt. Im Gegensatz zu einer Photodiode verfügt ein PSD jedoch über mehrere gegenüberliegende elektrische Kontakte. Dadurch kommt es zu einer Aufteilung des Photostromes auf die Kontakte, in Abhängigkeit von der Position des Lichtflecks. Die Position ermittelt man durch Bildung der Stromdifferenz zwischen zwei gegenüberliegenden Kontakten. Durch Normierung auf den Gesamtstrom wird das Positionssignal unabhängig von der einfallenden Lichtintensität. Ein eindimensionaler PSD erlaubt die kontinuierliche Positionsbestimmung eines Lichtflecks entlang einer Achse. Er verfügt über 3 Kontakte. Die Position ergibt sich aus:

$$X / L = l_{a1} - l_{a2} / l_{a1} + l_{a2} \tag{3.1}$$

Ein zweidimensionaler PSD hat zur Positionsbestimmung entlang von zwei Achsen 4 Kontakte. Die Position erhält man analog aus:

$$\begin{aligned} X / L &= l_{a1} - l_{a2} / l_{a1} + l_{a2} \\ Y / L &= l_{c1} - l_{c2} / l_{c1} + l_{c2} \end{aligned} \tag{3.2}$$

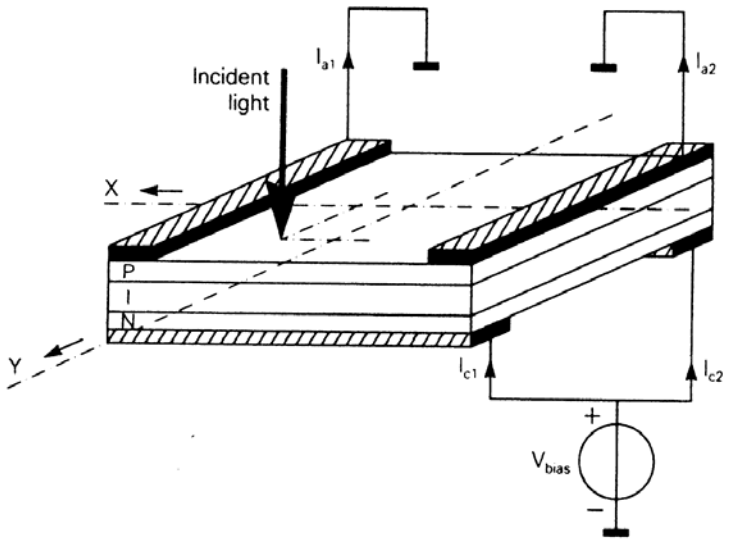


Abbildung 3.3: Zweidimensionaler PSD- Chip

Zu den PSD- Eigenschaften *gehören:*

- Hohe Linearität
- Geringe Temperaturdrift
- Hohe Empfindlichkeit
- Schnelle Anstiegszeiten

Da bei positionsempfindlichen Detektoren (PSD) im Gegensatz zu pixelelementverarbeitenden Kameras (CCD) die Auflösung nicht durch die Pixelgröße begrenzt wird und vor allem keine aufwendige Bildanalyse durch einen Rechner nötig ist, wurden zum Zwecke der visuellen Beobachtung am System zweidimensionale PSDs von SiTek ausgewählt. Unter Verwendung von zwei zweidimensionalen PSDs kann ein dreidimensionales Meßsystem realisiert werden. Dieses System verwendet hochgenaue zweidimensionale PSDs von Fa. SiTek wegen deren hoher Genauigkeit, ihrer Schnelligkeit und der geringen Nichtlinearität. Die PSDs werden in die PSD- Kameras eingebaut.

Zur Signalerfassung und Verarbeitung ist ein entsprechendes Verstärkersystem erforderlich, das die Strom-Messwerte von PSD in Positionsangaben umrechnet. Es wurde ein kostengünstiges Positionsmessverstärker OT-301 der Fa. SiTek ausgewählt.

Abbildung 3.4: Der Positionsmessverstärker OT-301

Die Positionsmessverstärker ermöglichen:

- Ein- oder zweidimensionale absolute optische Positionsmessung oder
- Präzise Zentrierung oder Nullabgleich bei Verwendung eines Quadrantendetektors.

Ein Vier-Kanal-Verstärkersystem verarbeitet mit Hilfe eines mathematischen Verfahrens den durch den LED im Detektor erzeugten Photostrom und erzeugt so eine analoge Ausgangsspannung für die X- und Y- Position, direkt proportional zum LED-Position und relativ unabhängig von Intensitätsschwankungen.
Der Positionsverstärker OT-301 bietet neben dem Positionsausgang für X und Y auch ein Summensignalausgang (S). Der OT-301 hat fünf Verstärkereinstellungen und kann somit einen Eingangsstrombereich von 0.1µA bis 2mA mit einer Modulationsfrequenz bis 15kHz verarbeiten (siehe Tabelle 1.1). Mit Hilfe der Null-Funktion kann der Anwender den Nullpunkt elektrisch an eine beliebige Position auf dem Positionsmessdetektor verschieben.

Transimpendanz-Verstärkung (V/A)	10^7 bis 10^3 Vier Eingangskanäle
Bereich Eingangsstrom	0.1µA bis 1,5mA
Ausgangsspannung	Position X,Y 10V Summensignal 0-10V
Detektor-Vorspannung	0V, 5V (Duolateral 0, 5, 10V)
Frequenzbereich	DC bis 15kHz, bereichabhängig
Bandbreite	$2{,}5\times10^{-4}$A/V 15kHz $6{,}2\times10^{-5}$A/V 1,5kHz $1{,}6\times10^{-5}$A/V 5kHz 3.9×10^{-6}A/V 1,25kHz $9{,}8\times10^{-7}$A/V 310Hz $2{,}5\times10^{-7}$A/V 80Hz
Ausgangsstecker	BNC
Eingangsstecker	Sub Min, D 9 PIN
Maße	14,15×6,4×21,59 cm

Tabelle 1.1: Technisch Spezifikationen des OT-301 Verstärkers

Die Verarbeitung von Sensorinformationen und die daraus abgeleitete intelligente Steuerung des Roboters sind komplexe Aufgaben, die mit den zur Zeit gegebenen technischen Möglichkeiten nur durch ein Mehrprozessorsystem zu bewältigen sind. Die einzelnen Prozessoren lassen sich dabei entsprechend ihren Aufgaben auf mehrere Hierarchie-Ebenen verteilen und müssen dabei unterschiedliche Anforderungen an die Rechenleistung, die Kommunikationsschnittstellen und die anzuschließende Peripherie erfüllen.
In der obersten Hierarchie-Ebene wird das Gesamtsystem dem Bediener zugänglich gemacht, d.h. der Roboter wird im Kontext mit den zu erledigenden Aufgaben bedienbar gemacht. Dafür ist ein standardmäßiger PC die beste Wahl, da hierfür vielerlei E/A- und Kommunikationsgeräte verfügbar sind. In der darunterliegenden Ebene werden die vom Bediener gegebenen Befehle in sensorgestützte Aktionen des Roboters umgesetzt. Die Beschreibung dieser Aktionen erfolgt in Form von Bewegungskommandos an die Teilsysteme des Roboters, z.B. an die mobile Plattform, an das Manipulations- oder das Sichtsystem. Für die dafür erforderliche Sensordatenverarbeitung und Bewegungssteuerung wird ein System aus vernetzten digitalen Signalprozessoren benutzt, dessen Gesamtrechenleistung durch Hinzunahme weiterer Prozessoren nach Bedarf erhöht werden könnte. In der untersten Ebene wird schließlich ein einfacher A/D- Wandler benutzt, an der über verschiedene Schnittstellen optische Sensoren angeschlossen werden.

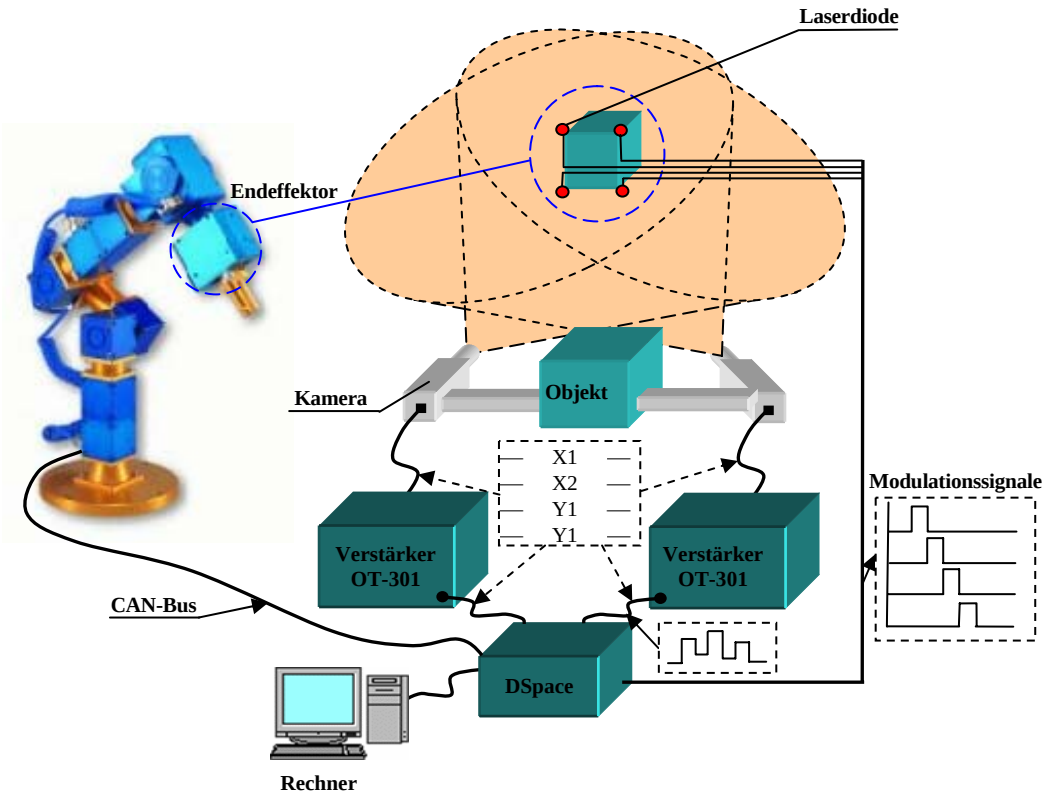


Abbildung 3.5: Die Hardwarerealisierung der gesamten optisch geführten Robotersteuerung

Unser Positionsmeßsystem besteht also aus vier Laserdioden, zwei PSD- Sensoren, zwei Verstärkern zur Signalerfassung und -verarbeitung sowie einem DSpace-Signalprozessorsystem für die Online-Berechnung und A/D-Umwandlung (siehe Abb. 3.5). Diese gesamte notwendige Hardware zur visuellen Beobachtung wurde erfolgreich installiert.

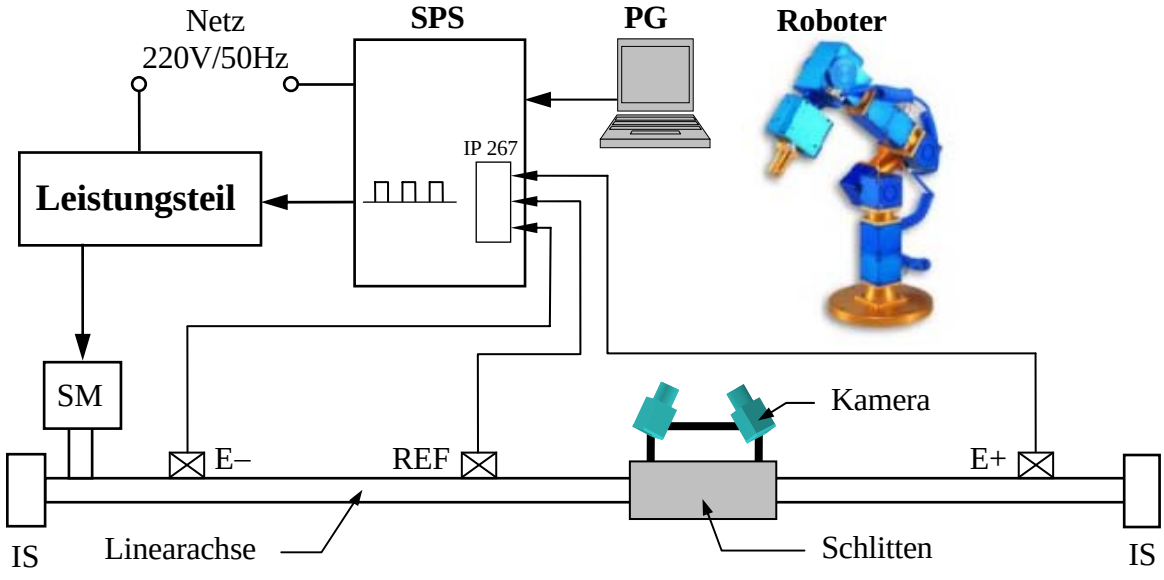
Um das Objekt mit den verschiedenen, definierten Geschwindigkeiten zu fahren, wurde ein entlang einer linearen Achse gesteuerter Schlitten verwendet (siehe Abb. 3.6). Dieser wird

durch eine speicherprogrammierbare Steuerung (SPS) entlang der *x*- Achse gesteuert. Dazu wurde ein System verwendet, das aus folgenden relevanten Einzelgeräten besteht:

- Programmiergerät PG 730
- Speicherprogrammierbare Steuerung SIMATIC S5 – 100U
- Schrittmotor-Positionier-Steuerung ELSTEP 2.D (Leistungsteil)
- Schrittmotor
- Linearachse mit zu positionierendem Schlitten
- Elektromagnetische Not-Endschalter und induktive Sensoren

Das leistungsfähige ***Programmiergerät PG 730*** gestattet das Eingeben, Kodieren, Testen und Dokumentieren von Programmbausteinen in Hochsprachen, nämlich als Funktionsplan FUP, Kontaktplan KOP oder Anweisungsliste AWL. Die Steuerungsangaben für die Positionieraufträge werden in der Programmiersprache STEP 5, und zwar in AWL, dargestellt.
Die ***speicherprogrammierbare Steuerung SIMATIC S5-100U*** ist durch folgende Charakteristika gekennzeichnet:

- Arbeit mit einem auf die Programmiersprache STEP 5 zugeschnittenen Wortprozessor.
- Nach der zeitlichen Abarbeitung der Steuerungsfunktionen handelt es sich um eine synchrone Steuerung, d.h. die Signalverarbeitung geschieht zeitgleich zu einem Taktssignal (im Gegensatz zur asynchronen Steuerung).
- Bezüglich der Arbeitsweise ist es eine Ablaufsteuerung, d.h. eine Steuerung, die schrittweise arbeitet. Das Weiterschalten auf den programmgemäß nächsten Schritt hängt von Weiterschaltbedingungen, den Transitionen, ab. Die Schritte werden von den induktiven Sensoren aktiviert. Es handelt sich also um eine prozessabhängige Ablaufsteuerung.



E-, E+ *Endschalter* ***REF*** *Referenzpunkt* ***PG*** *Programmiergerät*
IS *Impulssperre (Motor-Halt)* ***SM*** *Schrittmotor* ***SPS*** *Simatic S5- 100U*

Abbildung 3.6: Grobstruktur des Bewegungssystems

Die ***intelligente Peripheriebaugruppe IP 267*** verfügt über einen eigenständigen Mikrorechner und ist in ihrer Hardware zugeschnitten auf die Kommunikation mit den externen Sensoren und der CPU der SPS mittels internem BUS. Die IP 267 erzeugt Impulsfolgen, mit denen Schrittmotor- Leistungsteile angesteuert werden können, um die impulsgenaue, numerisch gesteuerte Positionierung und Bewegung des Schlittens zu realisieren. Über eigene Digitaleingänge kann die IP 267 Positioniervorgänge kontrollieren und bestimmte Ereignesse an die CPU melden.
Die ***Schrittmotor-Positionier-Steuerung ELSTEP 2.D*** (in Abb. 7.5 als Leistungsteil bezeichnet) ist das Bindeglied zwischen der IP 276 und dem Schrittmotor. Sie formt die energiearmen Impulse der IP 276 in eine Motorspannung von 40 V DC und einen Motorstrom von maximal 4 A um.
Der ***Schrittmotor*** ist ein 2 Phasen-Schrittmotor mit bipolarer Wicklung. Die Schrittmotorantriebe haben folgende Merkmale:

- Schrittgenaues Positionieren ohne Lageregelkreis
- Hohes Drehmoment bei kleinen Drehzahlen
- Großes Haltemoment im erregten Zustand
- Drehzahlsteuerung über die Schrittfrequenz
- Einfache und preiswerte Ansteuerungselektronik
- Schlupffreies Positionieren

Die Schrittmotoren können direkt mit binären Signalen (Impulsen) angesteuert werden. Die Schrittmotoren wandeln also elektrische Impulse, die von der Schrittmotor-Positionier-Steuerung kommen, in eine mechanische Drehbewegung mit definiertem Schrittwinkel um. Die Verfahrgeschwindigkeit des Schlittens wird mit der Impulsfrequenz festgelegt.
Auf der ***Linearachse*** (Alu-Profilschiene) wird ein mit Rollen gelagerter Schlitten verfahren. Auf dem Schlitten sind die zwei am Objekt befestigten Kameras (siehe dazu Abb. 2.3) montiert. Der Schlitten wird vom Schrittmotor über einen Zahnriemen angetrieben. Unterhalb des Schlittens befinden sich Metallstifte, die induktive Sensoren aktivieren. Mit Hilfe diesen Sensoren wird zuerst ein Referenzpunkt (Nullpunkt) ermittelt. Von diesem Punkt aus werden die notwendigen Bewegungen durchgeführt.

Kapitel 4

Trajektoriengenerierung

Ein weiteres Einsatzgebiet der optisch geführten Roboter bilden die bahngeführten Angaben (*Trajektoriengenerierung 1* und *2* in der Abb. 2.3), bei denen der Robotergreifer auf einer im allgemeinen im kartesischen Koordinatensystem definierten Trajektorie zu bewegen ist. Von der Art der Führungsgrößevorgabe und der Bahnregelung hängt ab, mit welcher Genauigkeit der Roboter die gewünschte räumliche Trajektorie abfährt. Eine geeignete Sollbahnvorgabe muss sowohl die von der Aufgabenstellung geforderte Trajektorie nachbilden als auch die kinematischen Eigenschaften der Robotermechanik berücksichtigen, damit die Bahnregelung später diesen Sollwerten folgen kann.
Das Hauptaugenmerk dieses Kapitels ist auf die zur Robotersteuerung gehörende Trajektoriengenerierung gerichtet. Unter eine Trajektorie versteht man dabei den zeitlichen Verlauf der Robotermatrix ${}^0T_n$ in kartesischen Koordinaten in Darstellung durch sogenannte homogene Koordinaten (siehe Abschnitt 6.1).

$$ {}^0T_n(t) = \begin{bmatrix} R_{3x3}(t) & p(t) \\ 0 & 1 \end{bmatrix}, \tag{4.1} $$

wo $R_{3x3}(t)$ den zeitlichen Verlauf der Orientierung des Endeffektors beschreibt, während $p(t)$ den Verlauf des Positionsvektors des Endeffektors repräsentiert.
Brady definiert in [9] eine Trajektorie als „… die zeitliche Aufeinanderfolge von Zwischenstellungen des Endeffektors zwischen Ausgangsposition P_0 und Endposition P_E … Die Kurve im Raum, die von Steuerzeichen verfolgt wird, heißt *Trajektorie*".
Die Trajektoriengenerierung hat die Aufgabe die Sollposition des Roboters während der Bewegung auf einer gewünschten Bahn im Abtastraster der Steuerung bereitzustellen. Voraussetzung für die Generierung einer Trajektorie ist, dass sie eindeutig beschrieben wird. Die Art der Bewegungsbahn des Endeffektors wird dabei durch jeweilige Aufgabe des Roboters bestimmt. Die wichtigsten Bewegungsarten lassen sich zusammenfassen zu:

- Bewegung von Punkt zu Punkt
- Bewegung auf Geraden
- Bewegung auf Raumkurven mit konstanter Geschwindigkeit
- Folgebewegungen
- Ausweichungen zur Vermeidung von Kollisionen

Als Solltrajektorie wird der gewünschte zeitliche Verlauf des Endeffektors des Roboters bezeichnet. Sie kann in analytischer Form als mathematische Funktion vorliegen und wurde dann mittels eines Algorithmus der Steuerung berechnet. Um den Rechenaufwand bei der

Generierung einer Trajektorie zu verringern, werden oft nur einfache Trajektorien, wie Geraden, Kreisbögen oder Parabeln in Betracht gezogen. Die Generierung der kartesischen Trajektorie kann als eine Interpolation zwischen zwei oder mehreren Bahnstützpunkten betrachtet werden. Für die in dieser Arbeit betrachteten Aufgaben reicht die Vorgabe von zwei Stützpunkten (Start- und Ziellage des Endeffektors) aus. Bei der Erweiterung auf Zirkular- und Parabelinterpolation wird die Angabe von mehreren Stütz- bzw. Hilfspunkten notwendig.
Damit ein Industrieroboter eine gezielte Bewegung durchführen kann, muss die Steuerung in der Lage sein, entsprechende Sollwerte für die Gelenkregelkreise zu generieren. Diese Aufgabe ist mehr oder weniger komplex, je nachdem ob der Roboter eine Punkt-zu-Punkt-Bewegung (PTP) durchzuführen hat oder der Endeffektor in Position und Orientierung einer definierten Bahn folgen soll (CP). Bei der PTP- Bewegung handelt es sich um eine unkoordinierte Achsbewegung, bei der alle Achsen unabhängig voneinander angesteuert werden. Die Roboterbewegungen im PTP- Modus sind immer dann sinnvoll, wenn der Endeffektor eine bestimmte Position anfahren soll, z.B. zum Greifen von Objekten, der Weg hin zu dieser Position jedoch nicht im einzelnen festgelegt sein muss. Die PTP-Bewegungen werden im allgemeinen so geplant, dass die Summe der Änderungen in den Gelenkfreiheitsgraden minimal ist, d.h. im Raum der Gelenkfreiheitsgrade (Konfigurationsraum) beschreibt eine Gerade den Weg zwischen Start- und Zielposition.
Bei der Durchführung komplexer Arbeitsaufgaben, wie z.B. Bahnschweißen, Entgraten, Kleberauftrag, muss dagegen die Roboterbewegung im Detail festgelegt sein. In diesen Fällen ist es wichtig, dass sich das mitgeführte Werkzeug mit unverändertem Abstand und Orientierung und mit vorgegebener Geschwindigkeit an dem betreffenden Werkstück entlang bewegt. Hierzu muss die gewünschte Trajektorie bekannt sein, d.h. zusätzlich zur gegebenen Geometrie der Bahn benötigt man für jeden Bahnpunkt die zugehörige Geschwindigkeit und Beschleunigung zur Generierung einer koordinierten Bewegung der Robotergelenke.
Ein weiteres wesentliches Ziel der Trajektoriengenerierung ist die Erzeugung mehrfach stetig differenzierbarer Bahnverläufe, um Sprünge in den Beschleunigungen und damit im Stellgrößenverlauf zu vermeiden. Hierdurch wird die Gefahr der Anregung von Strukturschwingungen der Roboterarme vermindert. Dies gilt auch für Bewegungen im PTP- Betrieb.
Die detaillierte Planung einer Trajektorie sollte soweit wie möglich automatisiert ablaufen, um die Programmierung des Roboters zu vereinfachen. Der Bediener gibt dem Modul für die Trajektoriengenerierung (s. Abb. 4.1) lediglich markante Stützpunkte zur Beschreibung des Bahnverlaufes sowie die Maximalwerte für Geschwindigkeiten und Beschleunigungen. Die Angabe der Stützpunkte erfolgt im Allgemeinen in kartesischen Koordinaten.
Das weitere Vorgehen richtet sich nach den Anforderungen an den Trajektorienverlauf:

1. Wird eine PTP-Bewegung ausgeführt, erfolgt zuerst eine inverse kinematische Transformation für die Zielkoordinaten, um die äquivalenten Gelenkwinkel zu berechnen. Die weitere Planung (Lineare Interpolation zwischen Start- und Zielpunkt, Berechnung des Geschwindigkeits- und Beschleunigungsprofils) findet in Gelenkkoordinaten statt (*Trajektoriengenerierung B* in der Abb. 4.1). Der Verlauf der kartesischen Bewegungen ist nicht ohne weiteres vorhersehbar (s. Abb. 4.2a).

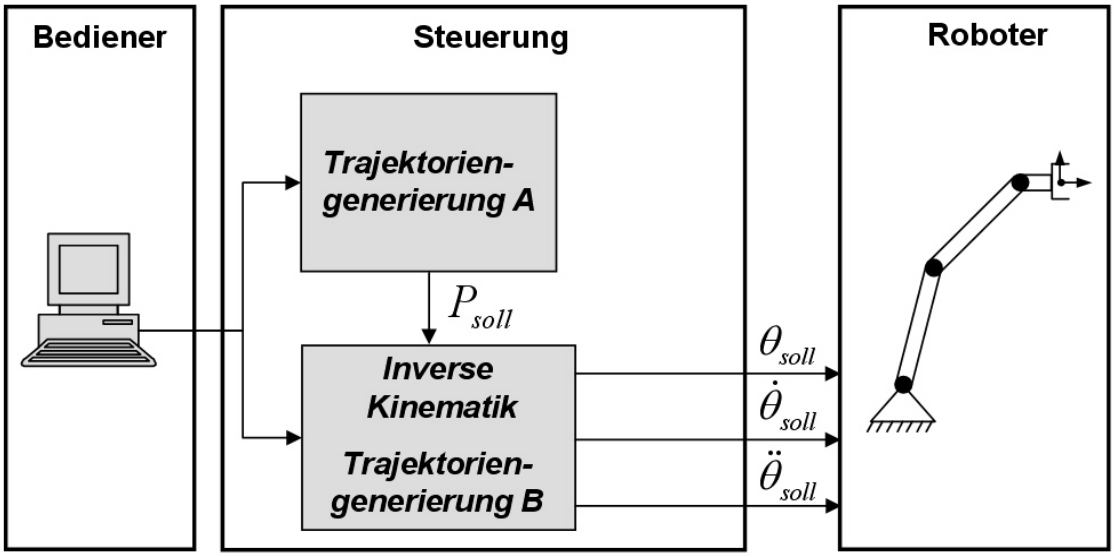


Abbildung 4.1: Anordnung der Modulen zur Trajektoriengenerierung in einer Robotersteuerung

1. Ist die Trajektorie in kartesischen Koordinaten geometrisch vorgeschrieben (CP-Bewegung), findet die Trajektorienberechnung auf kartesischer Ebene statt (*Trajektoriengenerierung A* in der Abb. 4.1). Die berechnete Trajektorie wird in jedem Interpolationstakt mit Hilfe der inversen Kinematik in Gelenkkoordinaten transformiert (s. Abb. 4.2b). Da die hierzu notwendige inverse Kinematik sehr aufwendig sein kann, werden Steuerungen mit leistungsfähigeren Prozessoren benötigt.

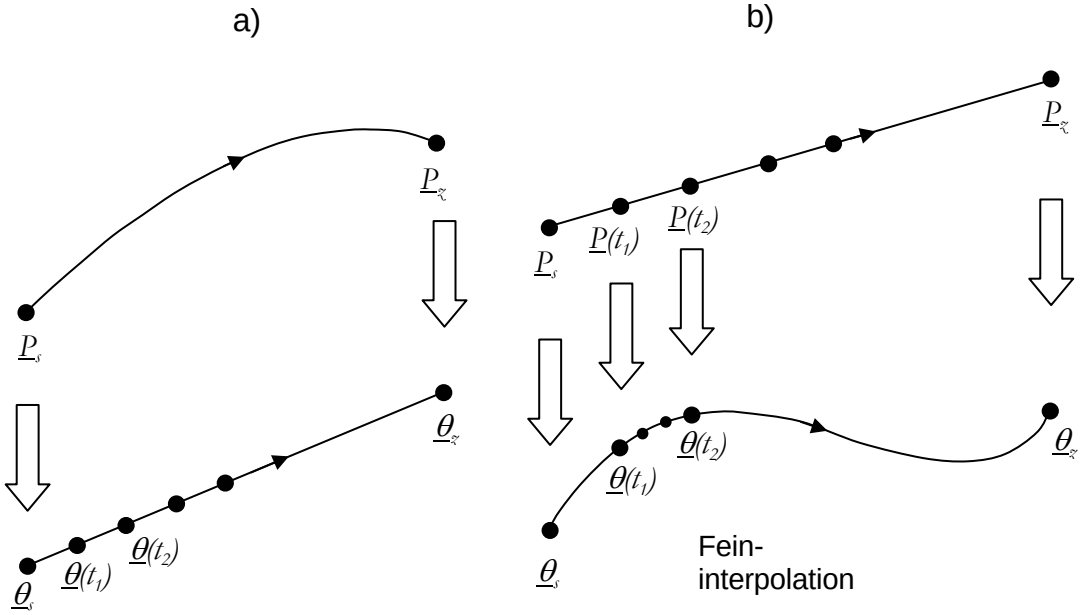


Abbildung 4.2: Prinzipielle Vorgehensweise bei der Trajektoriengenerierung für PTP- Bewegungen (a) und CP- Bewegungen (b)

Man kann grundlegend zwischen zwei verschiedenen Arten der kartesischen Steuerung unterscheiden: eine Steuerung, bei der die Berechnung der Trajektorien und der inversen Kinematik vor der Bewegung durchgeführt werden, und eine zweite, bei der die Transformationen der inversen Kinematik und die Trajektoriengenerierung unter

Echtzeitbedingungen, d.h. während der Bewegung des Roboters, ablaufen müssen (s. Abb. 4.1).
Bei der Online-Steuerung (Echtzeit) müssen die Berechnungen der Trajektorienpunkte sowie die Transformation in Gelenkkoordinaten durchgeführt werden, während der Roboter sich bewegt. Dies bringt die Notwendigkeit mit sich, diese Berechnungen sehr schnell auszuführen. Dies hat dann den Vorteil, dass die Trajektoriengenerierung in kartesischen Koordinaten durchgeführt werden kann, was bei der optisch geführten Robotersteuerung die Voraussetzung ist.
Bei der Offline-Steuerung ist die grundlegende Voraussetzung, dass die abzufahrende Trajektorie schon vor dem Beginn der Bewegung genau festgelegt ist. So steht dem Rechner ausreichend Zeit zur Verfügung, um ihre Realisierung detailliert vorzugeben. Soll die Bewegung später ausgeführt werden, liest der Rechner einfach in jedem Abtastintervall einen Vektor aus dem Speicher aus, dessen Komponenten die verschiedenen Gelenkwinkel sind. Diese werden den Roboterachsen als Sollwerte zur Verfügung gestellt.
Da die Trajektoriengenerierung durch Offline- Steuerung für die optisch geführten Robotersteuerung offensichtlich nicht verwendet werden kann, werden zukünftig nur die Methoden zur Berechnung der Trajektorien für die Online- Steuerung in Betracht gezogen.
Ein wesentlicher Gesichtspunkt bei der Generierung geometrischer Bahnen in kartesischen Koordinaten ist die Durchführbarkeit der Bewegung. Dabei können folgende Probleme auftreten:

1. Unerreichbare Bahnpunkte:
   Obwohl Start- und Zielpunkt innerhalb des erreichbaren Arbeitsraumes liegen, befindet sich ein Teil der gegebenen Bahn außerhalb dieses Bereiches. Die Abbildung 4.3 zeigt dies am Beispiel einer geradlinigen kartesischen Bewegung eines Roboters mit zwei Rotationsachsen.

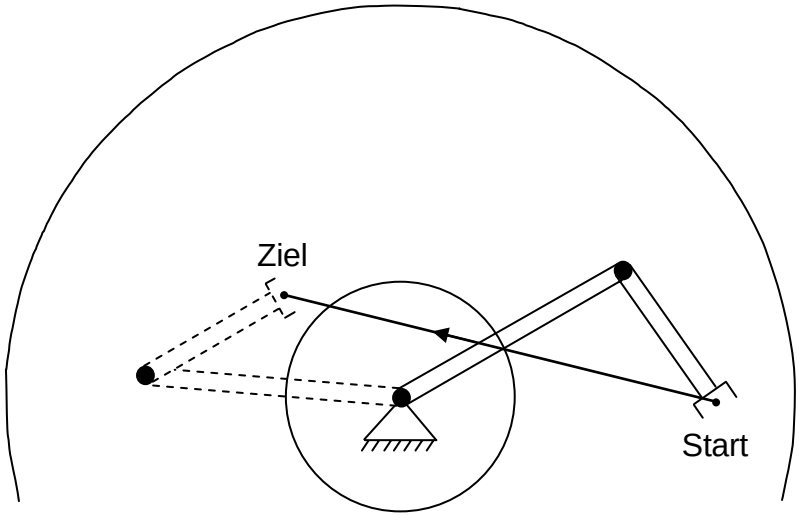


Abbildung 4.3: Zur Problematik unerreichbarer Bahnpunkte

2. Mechanische Begrenzung der Gelenkbewegungen:
   Jeder Roboter hat, bedingt durch seinen mechanischen Aufbau, Begrenzungen in der Bewegungsmöglichkeit seiner Gelenke. Dies kann dazu führen, dass bei der Anwendung der inversen Kinematik, die im allgemeinen mehrere Lösungen ergibt, Lösungszweige

existieren, die Gelenkwinkel jenseits der Begrenzungen enthalten und damit keine realisierbare Bewegung zwischen Start- und Zielposition beschreiben.

3. Nicht angepasste Bahngeschwindigkeit bzw. -beschleunigung:
   Für die Berechnung einer Trajektorie entlang der gewünschten kartesischen Bahn ist die Vorgabe von Maximalwerten für die Bahngeschwindigkeit und -beschleunigung notwendig. Diese müssen so gewählt werden, dass die resultierenden Geschwindigkeits- und Beschleunigungswerte für die Gelenke das Leistungsvermögen der Antriebe nicht überfordern.

Wie schon oben erwähnt wurde, beschränkt sich die Planung des Roboterbahnes nicht nur auf den geometrischen Verlauf der Trajektorie, sondern es müssen auch zeitliche Verläufe berücksichtigt werden. Bei der Generierung des Geschwindigkeits- bzw. des Beschleunigungsprofils wird der Weg als zusätzlicher Bahnparameter mitberücksichtigt. Die erste Ableitung dieses Weges entspricht der Bahngeschwindigkeit, die zweite der Bahnbeschleunigung und die dritte dem Ruck.
Man plant also den Weg vernünftigerweise so, dass der Endeffektor sanft beschleunigt wird. In der Abbildung 4.4 ist eine Gerade im Raum zwischen Solllage (Ziel) und der Istlage (Start) gezeichnet. Der Weg wird jedoch ungleichmäßig auf Zeitintervalle aufgeteilt, da der Roboter sich nicht von Anfang an mit der Maximalgeschwindigkeit bewegen kann. Es ergibt sich eine typisch parabolische Kurve im Weg/Zeit- Diagramm. Abbildung 4.4 zeigt einen solchen Weg und spezielle diskrete Zeitpunkte in einem *x-y* Koordinatensystem.

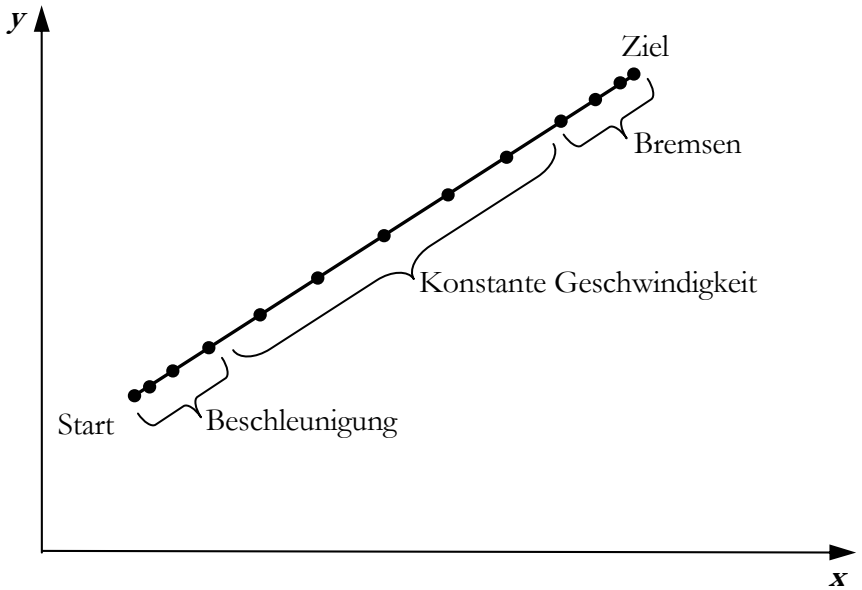


Abbildung 4.4: Die Trajektorie eines Endeffektors in der x- und y- Koordinaten

Die vorgegebenen Stützpunkte (Start- und Zielpunkte) können wahlweise entweder linear oder parabelförmig zur Solltrajektorie miteinander verbunden werden. Nur zu Beginn und am Ende der Bewegung sind Beschleunigungen zugelassen. Diese dürfen jedoch die festgelegten Begrenzungen nicht überschreiten. Die restliche Bahn soll konstant mit einer möglichst großen Geschwindigkeit abgefahren werden. Zur Optimierung des Geschwindigkeitsprofils hinsichtlich einer Durchlaufzeit der Trajektorie sollen die Beschleunigungen mit dem zugelassenen Ruck erfolgen. Zur Vermeidung von Schwingungen in der Robotermechanik wird großer Wert auf einen sanften Verlauf der Trajektorie gelegt. Somit besteht die Forderung nach stetiger Bahngeschwindigkeit und –beschleunigung sowie nach einer

geometrischen Kontur, die weder einen Knick noch eine Krümmungsänderung aufweisen darf. Außerdem müssen sämtliche vom Anwender vorgegebenen Begrenzungen eingehalten werden.
Häufig wird bei Robotersteuerungen von einem trapezförmigen Geschwindigkeitsprofil ausgegangen, bei dem sich die Geschwindigkeit linear mit der Zeit ändert. (s. Abb. 4.5).

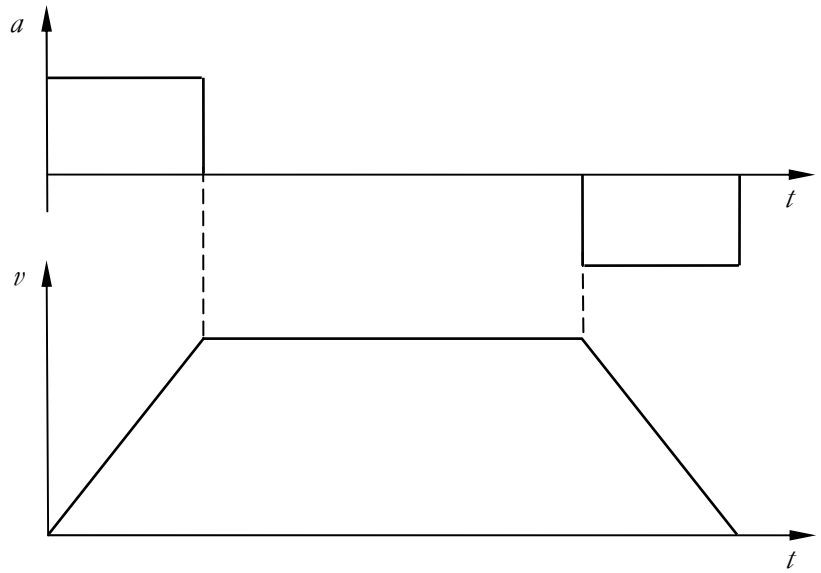


Abbildung 4.5: Trapezförmiges Geschwindigkeitsprofil mit dazugehörigem zeitlichem Verlauf der Beschleunigung

Dabei beschleunigt der Endeffektor konstant bis auf eine feste Geschwindigkeit und bremst bis zum Stillstand am Zielpunkt ab.
Um jedoch die geforderte Stetigkeit in der Trajektorienbeschleunigung zu erzielen, bedarf es eines nichtlinearen Geschwindigkeitsprofils. Integriert man den in Abhängigkeit der Zustandswerte Beschleunigung a, Geschwindigkeit v und Weg s sowie der Begrenzungen für Geschwindigkeit v_{max} und Beschleunigung a_{max} unstetig vorgegebenen Ruck $r=da/dt$ dreifach nach der Zeit, so erhält man eine lineare Funktion für die Beschleunigung, Parabeln zweiter Ordnung für die Geschwindigkeit sowie Parabeln dritter Ordnung für den Weg (s. Abb. 4.6). Die Symmetrie zur halben Verstellzeit im zeitlichen Verlauf der Trajektoriengeschwindigkeit und –beschleunigung (s. Abb. 4.6) resultiert daraus, dass am Trajektorienanfang und –ende Stillstand vorliegen muss, während dazwischen mit einer konstanten Geschwindigkeit fahren wird.

In vielen Robotersteuerungen sind üblicherweise Algorithmen zur Berechnung von geraden und kreisförmigen Trajektoriensegmenten (Linear- und Zirkularinterpolation) implementiert, die als Grundelemente zur Erzeugung von kartesischen Trajektorien dienen. Wie im Folgenden gezeigt wird, ergeben sich an den Übergängen zwischen solchen Trajektoriensegmenten im Allgemeinen unerwünschte Unstetigkeiten im Trajektorienverlauf bzw. in dessen Ableitungen. Deshalb ist es sinnvoll, als drittes Grundelement Polynomfunktionen zu verwenden, die je nach Anforderung an die Randwerte der Ordnung drei bis fünf aufweisen müssen. Im Folgenden werden nur die Polynomfunktionen zur Beschreibung kartesischer Bahnen detailliert beschrieben. Weitere Informationen zu diesem Thema sind bei [9] zu finden.

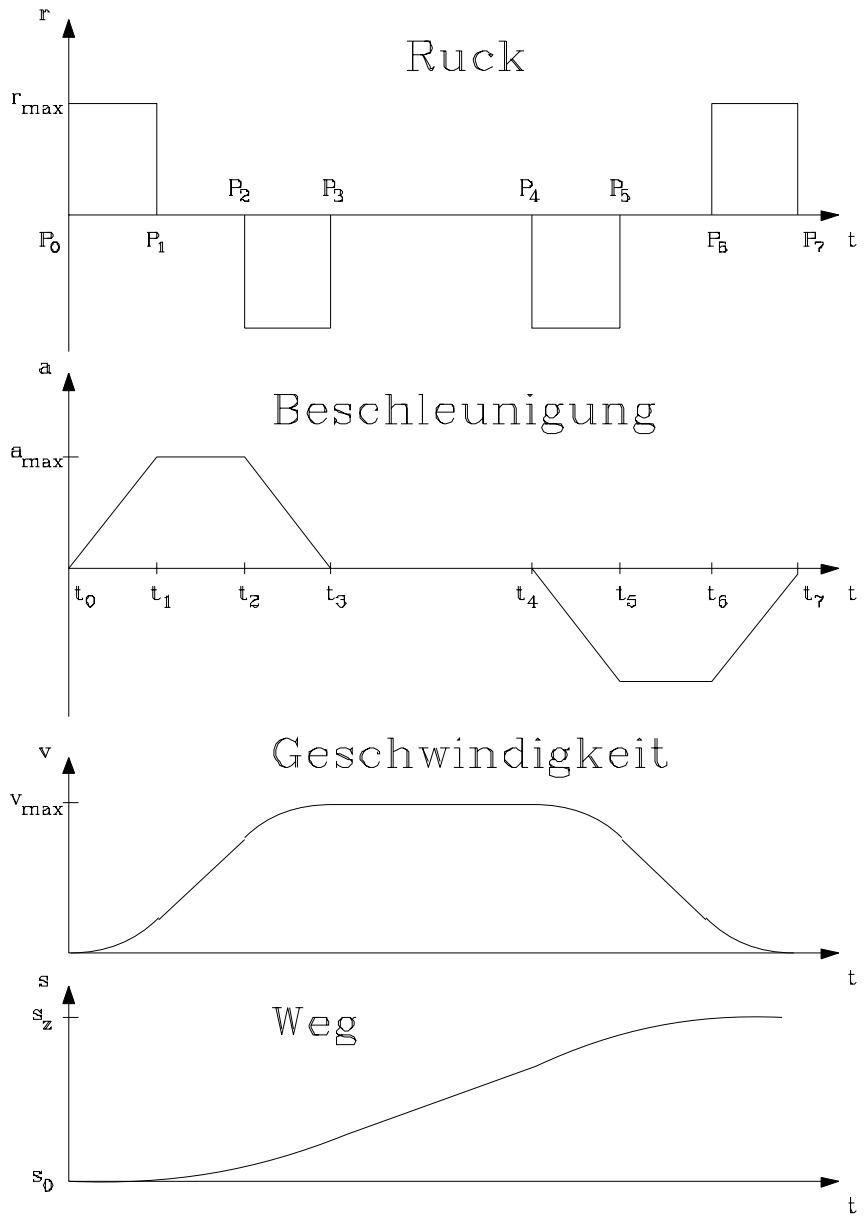


Abbildung 4.6: Zeitlicher Verlauf von Ruck, Beschleunigung, Geschwindigkeit und Weg

Wenn man eine Trajektorie von Punkt P_0 nach Punkt P_E beschreiben will, dann ist für den gesamten Verlauf der Bewegung auch $P(t)$, $\dot{P}(t)$, $\ddot{P}(t)$ zu spezifizieren. Dabei ist sicherzustellen, dass zu jedem Zeitpunkt *t* gilt

$$\dot{P}(t) = \frac{dP}{dt}, \qquad \ddot{P}(t) = \frac{d\dot{P}}{dt}, \tag{4.2}$$

dass also der eine Verlauf immer gerade die Ableitung des anderen dargestellt.
Eine verbreitete Variante zur Lösung dieses Problems sind Polynomansätze [28]. Dabei gibt man den Verlauf der Lage über der Zeit als Polynom in *t* vor: *P(t)*. Die Ableitungen dieses Polynoms nach t sind leicht gebildet und ergeben dann den Verlauf der anderen Größen $\dot{P}(t)$ und $\ddot{P}(t)$.
Die Verwendung von Polynomen als elementare Funktionen zur Generierung von Trajektoriensegmenten gestattet eine hohe Flexibilität beim Anschluss an benachbarte

Segmente in Bezug auf die Wahl der Randbedingungen. Hierdurch wird insbesondere die Schaffung der geforderten stetigen Übergänge bis zur zweiten Ableitung zwischen den Bahnsegmenten ermöglicht.
Die Ordnung des Polynoms richtet sich nach der Anzahl der zu berücksichtigenden Randbedingungen am Anfang und Ende der Funktion. Bei der Trajektorienberechnung ist je nach Anwendungsfall eine Ordnung des Polynoms zwischen 3 und 5 notwendig. Im Allgemeinen sieht die Polynomfunktion so aus:

$$F(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + \cdots + a_n t^n , \tag{4.3}$$

wo a_0, a_1, …, a_n die Koeffizienten des Polynoms sind.
Zu den Koeffizienten des Polynoms kommt man über die Bedingungen, die man an die Trajektorie stellt. Gibt man z.B. ein Polynom 5. Ordnung vor,

$$P(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 , \tag{4.4}$$

dann gilt es sechs Koeffizienten zu bestimmen. An die gewünschte Trajektorie können also sechs Bedingungen gestellt werden. Zusätzlich zu Anfangs- und Endpunkt ist die Auswahl von Anfangs- und Endgeschwindigkeit sowie -beschleunigung möglich. Für einfache geradlinige Bewegungen setzt man die letzteren Start und Endwerte zu null.

$$\dot{P}(0) = 0, \qquad \dot{P}(t_e) = 0 , \qquad \ddot{P}(0) = 0, \qquad \ddot{P}(t_e) = 0 . \tag{4.5}$$

Setzt man Anfangs- und Endpunkt P_0, P_E in (4.5) in die Ableitungen von (4.4)

$$\begin{aligned} \dot{P}(t) &= a_1 + 2a_2 t + 3a_3 t^2 + 4a_4 t^3 + 5a_5 t^4 \\ \ddot{P}(t) &= 2a_2 + 6a_3 t + 12a_4 t^2 + 20a_5 t^3 \end{aligned} \tag{4.6}$$

ein, dann ergibt sich ein Gleichungssystem zur Bestimmung der Koeffizienten a_0 ... a_5. In [28] wurden diese Koeffizienten aus den Gleichungen (4.4) und (4.6) wie folgt bestimmt:

$$\begin{aligned} a_0 &= P_0 \\ a_1 &= \dot{P}_0 \\ a_2 &= \frac{\ddot{P}_0}{2} \\ a_3 &= \frac{20P_e - 20P_0 - (8\dot{P}_e + 12\dot{P}_0)t_e - (3\ddot{P}_0 - \ddot{P}_e)t_e^2}{2t_e^3} \\ a_4 &= \frac{30P_0 - 30P_e + (14\dot{P}_e + 16\dot{P}_0)t_e + (3\ddot{P}_0 - 2\ddot{P}_e)t_e^2}{2t_e^4} \\ a_5 &= \frac{12P_e - 12P_0 - (6\dot{P}_e + 6\dot{P}_0)t_e - (\ddot{P}_0 - \ddot{P}_e)t_e^2}{2t_e^5} \end{aligned} \tag{4.7}$$

Beschreibt man die Lage des Endeffektors (Position und Orientierung) durch einen Vektor $\underline{P}=(x, y, z, a, \beta, \gamma)^T$ im kartesischen Referenzsystem, so ergibt sich bei der Vorgabe einer geometrischen Bahn als Funktion der Lage eine implizite Abhängigkeit von der Zeit in der Form:

$$\underline{P} = \underline{P}(t) = \begin{pmatrix} x(t) \\ y(t) \\ z(t) \\ \alpha(t) \\ \beta(t) \\ \gamma(t) \end{pmatrix} \tag{4.8}$$

Bei der Anwendung von Polynomfunktionen 5. Ordnung im kartesischen Koordinatensystem muss für jede der maximal 6 Koordinaten ein Polynom generiert werden.

$$\begin{aligned} x(t) &= a_{0x} + a_{1x}t + a_{2x}t^2 + a_{3x}t^3 + a_{4x}t^4 + a_{5x}t^5 \\ y(t) &= a_{0y} + a_{1y}t + a_{2y}t^2 + a_{3y}t^3 + a_{4y}t^4 + a_{5y}t^5 \\ z(t) &= a_{0z} + a_{1z}t + a_{2z}t^2 + a_{3z}t^3 + a_{4z}t^4 + a_{5z}t^5 \\ \alpha(t) &= a_{0\alpha} + a_{1\alpha}t + a_{2\alpha}t^2 + a_{3\alpha}t^3 + a_{4\alpha}t^4 + a_{5\alpha}t^5 \\ \beta(t) &= a_{0\beta} + a_{1\beta}t + a_{2\beta}t^2 + a_{3\beta}t^3 + a_{4\beta}t^4 + a_{5\beta}t^5 \\ \gamma(t) &= a_{0\gamma} + a_{1\gamma}t + a_{2\gamma}t^2 + a_{3\gamma}t^3 + a_{4\gamma}t^4 + a_{5\gamma}t^5 \end{aligned} \tag{4.9}$$

Zur Berechnung der maximal 36 Koeffizienten müssen demnach für jede Koordinate die Randbedingungen entsprechend der Gleichung (4.5) festgelegt sein. Die Vorgehensweise zur Bestimmung der Koeffizienten des Polynoms wurde schon oben beschrieben.
Wie man aus der Gleichung (4.8) sieht, reicht die Angabe von Start- und Zielpunkt nicht aus, da auch die Orientierungsänderung des Endeffektors von Bedeutung ist. Gemäß der Gleichungen (4.9) für die Orientierung (α, β, γ) wird die Endeffektorstellung gleichmäßig während der Trajektorienverlauf von der Anfangsorientierung in die Endorientierung geändert (s. Abb. 4.7).

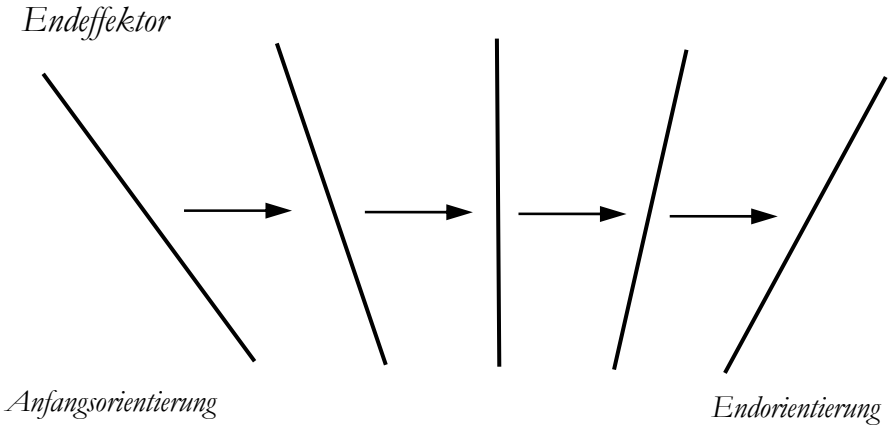


Abbildung 4.7: Linearbewegung mit variabler Orientierung

In Abbildung 4.8 sind die gemäß den Polynomgleichungen (4.9) berechnete Trajektorien im 3D-Raum dargestellt. Innerhalb von fünf Sekunden verändert sich die Position des Endeffektors von der Anfangs- in die Endposition. Die Beschleunigung und Geschwindigkeit beginnen und beenden bei null. Der Positionsverlauf folgt einem Polynom 5. Ordnung in *t*.

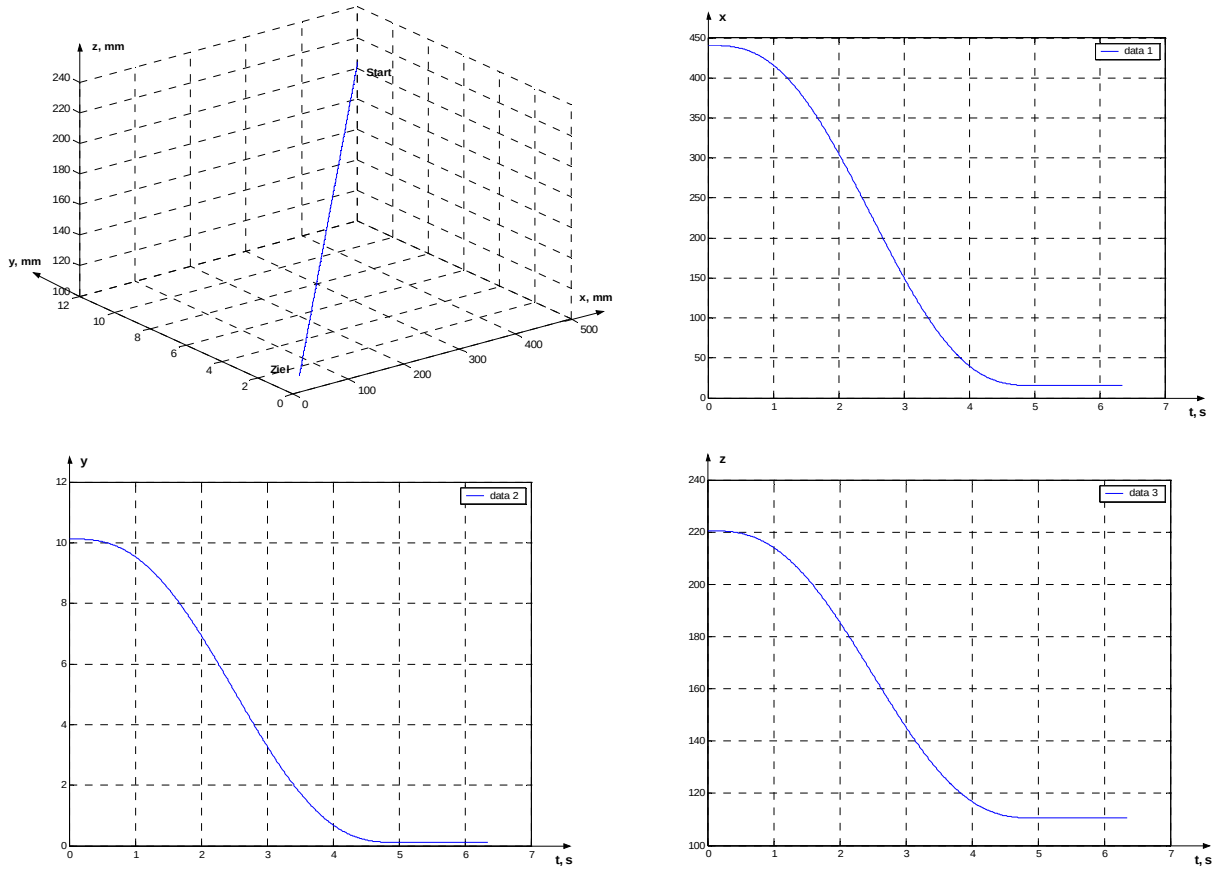


Abbildung 4.8: Verlauf des Endeffektors im 3D-Raum und entsprechende Trajektorienverlauf für jede Koordinate

In Abbildung 4.9 ist ein zeitlicher Verlauf von Ruck, Beschleunigung, Geschwindigkeit und Weg für die generierte Trajektorie gemäß des Polynoms 5. Ordnung dargestellt. Die Verläufe der Geschwindigkeit und der Beschleunigung besitzen Stetigkeit, d.h. weisen keine sprunghaften Veränderungen von Geschwindigkeit, Beschleunigung und Ruck auf.
Trajektorien dieser Art werden für die Blöcke *Trajektoriengenerierung 1* und *2* des Gesamtsystems der optisch geführten Robotersteuerung (siehe Abb. 2.3) mit Hilfe des Programms „trajgen.c“ (siehe Anhang A) generiert. Dabei werden für einen vorgegebenen relativen Abstand zwischen dem Objekt und Endeffektor mit Hilfe des Blocks *Trajektoriengenerierung 1* räumliche Trajektorien gestaltet. Trajektorien dieser Art werden sowohl für die simulativen und experimentellen Untersuchungen der direkten und inversen Kinematik als auch für die Kamerakalibrierung und Verifikationen der gesamten optisch geführten Robotersteuerung, die im folgenden Kapiteln betrachtet werden, benutzt.

Aufgrund des Differenzsignals $\Delta\vec{P}_o$ in Abb. 2.3, das sich sprunghaft ändert, da das Kameramodell viel langsamer (100ms) als alle anderen Elemente der Robotersteuerung (10ms) getaktet ist (siehe Kapitel 6), werden im Block *Trajektoriengenerator 2* sanfte Trajektorien nach Abb. 4.8 und 4.9 generiert, um die ruckfreie Objektverfolgung zu ermöglichen.

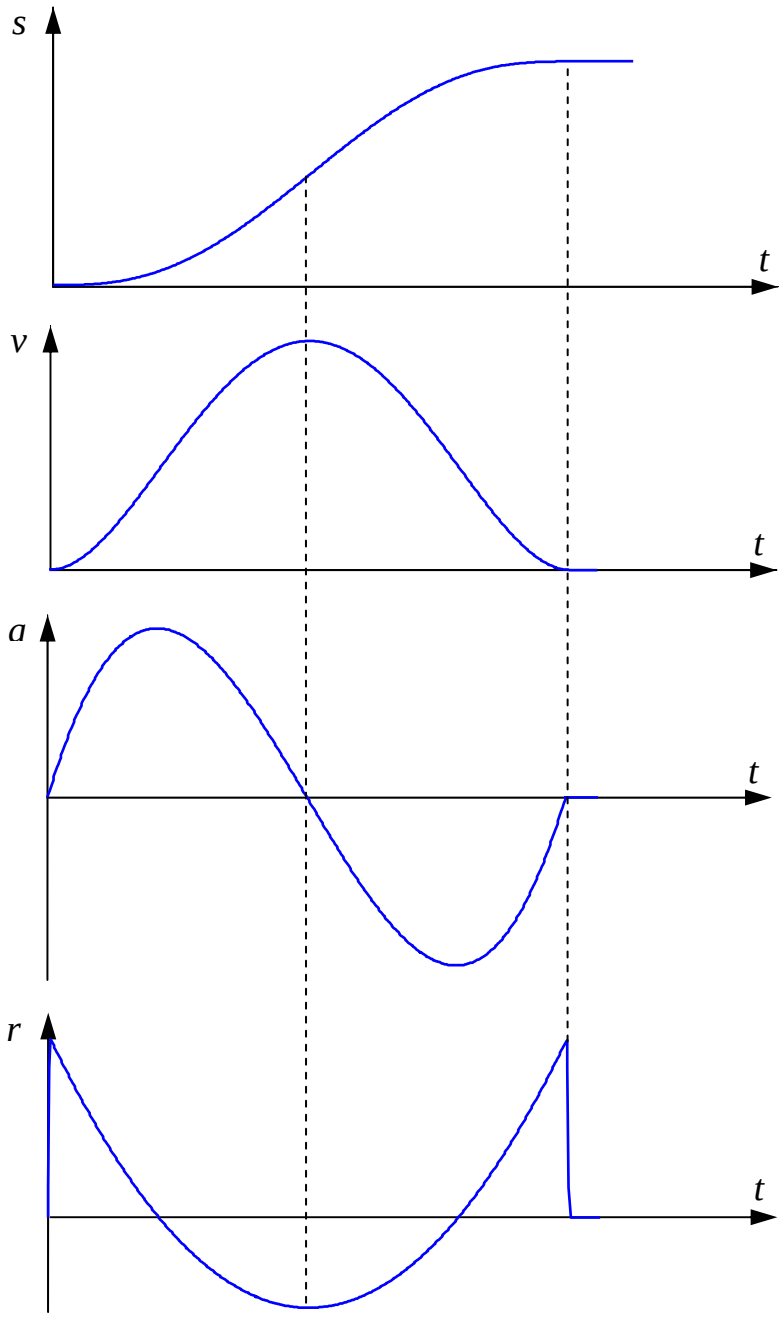


Abbildung 4.9: Zeitlicher Verlauf von Ruck, Beschleunigung, Geschwindigkeit und Weg für die Polynomfunktion der 5. Ordnung

Kapitel 5

Kinematik des Roboters *amtec r7*

Wie man aus dem Signalflussbild der visuellen Robotersteuerung (Abb. 2.3) sieht, spielt die Kinematik des Roboters mit deren Nichtlinearität und Mehrdeutigkeit für die Funktion der Steuerung eine wesentliche Rolle. Eine *kinematische* Beschreibung dient ausschließlich dazu, die relative Lage von aufeinander folgenden Gliedern zu beschreiben. Eine solche Darstellung vernachlässigt bewusst mechanische Eigenschaften, wie zum Beispiel das wirkliche Aussehen oder die Verformungen der einzelnen Manipulatorglieder. Als Kinematik bezeichnet man die Betrachtung von Bewegungen, ohne Berücksichtigung der Kräfte (Dynamik), die sie verursachen. Bei Robotern geht es dabei insbesondere um die Beziehung zwischen den Angaben der Winkelkoordinaten der Antriebsachsen und den kartesischen Koordinaten speziell des Endeffektors. Das Endziel der Robotersteuerung besteht im Erreichen einer bestimmten Lage des Endeffektors im Raum. Seine Lage ist gekennzeichnet durch seine Position und Orientierung im Bezugskoordinatensystem, dem globalen Koordinatensystem des Arbeitsraumes. Ein typischer Knickarm-Roboter, an dessen Ende sich der Endeffektor befindet, ist aus mehreren, über Drehgelenken miteinander verbundenen Gliedern, aufgebaut. Um den Endeffektor an eine bestimmte Stelle, deren Position und Orientierung in kartesischen Koordinaten vorgegeben ist, zu bewegen, sind die Positionswerte der einzelnen Gelenke inkrementell zu verändern. Es ist also erforderlich, die kartesischen Koordinaten des Endeffektors aus den Achskoordinaten und umgekehrt zu bestimmen (siehe Abb. 5.1).

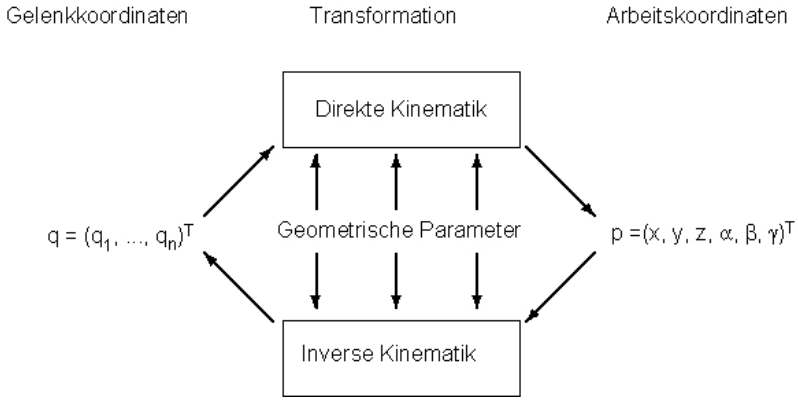


Abbildung 5.1: Problematik in der Kinematik

Die Transformation von Achs- bzw. Gelenkkoordinaten in kartesischen Koordinaten wird als *direkte Kinematik* bezeichnet. Hierbei wird aus den vorhandenen Positionswerten der Gelenke die Lage des Endeffektors, also seine Position und Orientierung im globalen Koordinatensystem des Arbeitsraumes bestimmt. Die Transformation von kartesischen Koordinaten in die Gelenkkoordinaten des Roboters wird als *inverse Kinematik* bezeichnet.

Die inverse Kinematik stellt das weitaus komplizierte Problem dar, da hierbei ein System, bestehend aus mehreren nichtlinearen, transzendenten Gleichungen, zu lösen ist. In diesem Kapitel wird die Kinematik für den 7-achsigen Roboter *amtec r7*, der bereits im Kapitel 3 ausführlich beschrieben wurde, betrachtet.

5.1 Direkte Kinematik

5.1.1 Grundlagen

Die direkte Kinematik eines Roboters, die in [28] und [70] ausführlich beschrieben wurde und in diesem Abschnitt kurz erläutert wird, ordnet einer bestimmten Stellung seiner Gelenke, einer Konfiguration, einen Punkt im kartesischen Arbeitsraum zu. Betrachtet man bei einem Roboter mit *n* Gelenken die Gelenkwinkel als Koordinaten in einem *m*-dimensionalen Raum, dann ist die direkte Kinematik eine Transformation dieser Koordinaten in den kartesischen Raum in Bezug auf das Basiskoordinatensystem des Roboters. Da dieser Stellung des Roboters genau ein Punkt im Arbeitsraum entspricht, stellt die direkte Kinematik eine eindeutige Beziehung dar. Diese Aussage wird später in diesem Abschnitt mathematisch bekräftigt. Ein Verfahren, um die kinematischen Beziehungen zwischen den einzelnen Gliedern des Roboters zu beschreiben, ist die von Denavit und Hartenberg entwickelte Methode unter Verwendung der nach ihnen benannten Denavit-Hartenberg (DH) – Parameter [28], [70]. Für jedes Gelenk gibt es ein zugehöriges Koordinatensystem. Die DH-Parameter beschreiben effizient die Transformationen von einem Gelenkkoordinatensystem zum nächsten, wobei die Transformationen zwischen den einzelnen Gelenken durch Verkettung der entsprechenden Transformationen entstehen.

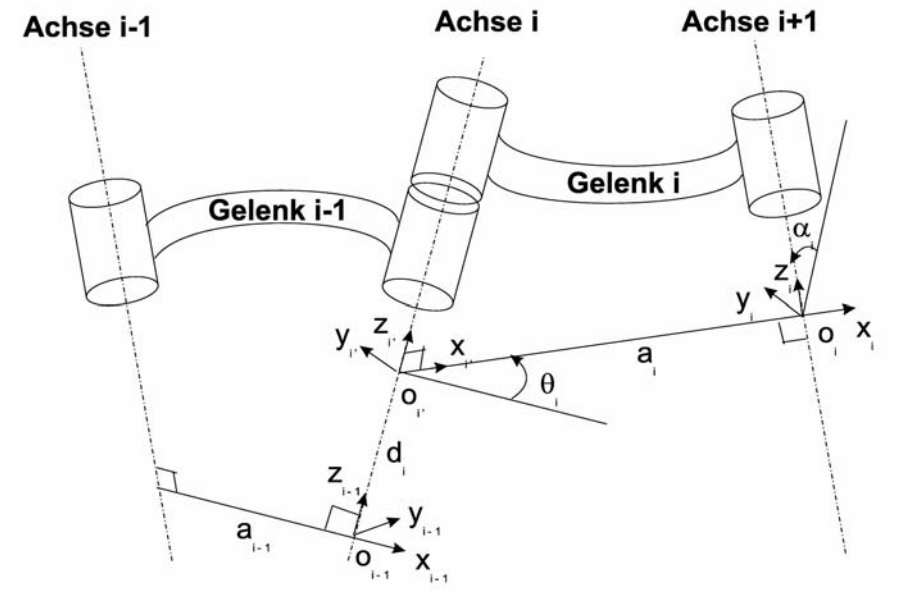


Abbildung 5.2: Grafische Darstellung der Denavit-Hartenberg-Parameter

Zur Abbildung der kinematischen Relation zwischen zwei Gelenken sind die Koordinatensysteme und die entsprechenden Parameter in Abb. 5.2 grafisch dargestellt. Dabei lassen sich die kinematischen Denavit-Hartenberg- Parameter in folgender Weise definieren [28]:

θ_i : Drehwinkel von der X_{i-1}-Achse zur X_i-Achse um die Z_{i-1}-Achse

α_i : Drehwinkel von der Z_{i-1}-Achse zur Z_i-Achse um die X_i-Achse

a_i : Abstand vom Schnittpunkt der Z_{i-1}-Achse und die X_i-Achse zum Ursprung des i-ten Koordinatensystems entlang der X_i-Achse oder der kürzeste Abstand zwischen der Z_{i-1}-Achse und Z_i-Achse

d_i : Abstand des Ursprungs des (i-1)-ten Koordinatensystems zum Schnittpunkt von der Z_{i-1}-Achse und X_i-Achse entlang der Z_{i-1}-Achse

Mit Hilfe der Denavit-Hartenberg- Parameter lässt sich damit eine Transformationsmatrix formulieren, mit der ein Achskoordinatensystem (*i-1*) in ein benachbartes Achskoordinatensystem (*i*) durch zwei Drehungen und zwei Verschiebungen überführt werden kann:

1. eine Rotation mit dem Winkel θ_i um die z_{i-1}- Achse, mit dem Ziel, die x_{i-1}- Achse parallel zur *x*- Achse auszurichten.
2. eine Translation um d_i entlang der z_{i-1}- Achse zu dem Punkt, wo sich die z_{i-1}- und x_i- Achse schneiden.
3. eine Translation um a_i entlang der x_i- Achse, um die Koordinatenursprünge zur Deckung zu bringen.
4. eine Rotation mit dem Winkel α_i um die x_i- Achse mit dem Ziel, die *z*- und *y*- Achsen zur Übereinstimmung zu bringen.

Diese vier Operationen werden in einer *4x4* Transformationsmatrix ${}^{i-1}T_i$, der sogenannten Denavit-Hartenberg- Matrix zusammengefasst:

$$ {}^{i-1}T_i = rot(z_{i-1},\theta_i)\,trans(z_{i-1},d_i)\,trans(x_i,a_i)\,rot(x_i,\alpha_i), \tag{5.1}$$

die ausmultipliziert die Transformationsmatrix

$$ {}^{i-1}T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.2}$$

ergeben, wobei

α_i , a_i , d_i = konstante Parameter,
θ_i = Gelenkvariable sind.

Die Transformationsmatrizen ${}^{i-1}T_i$ zur Transformation von einem Gelenk-Koordinatensystem *(i-1)* zum nächsten *(i)* in der Gleichung (5.2) lassen sich auch in anderer Form aufschreiben:

$$ {}^{i-1}T_i = \begin{pmatrix} R & P \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \tag{5.3}$$

wobei die Rotationsmatrix *R* und Translationsvektor *P* auftreten.

Für einen Roboter mit *n*-starren- Gelenken lässt sich die Transformation von erstem Gelenk *(0)* zu letztem Gelenk *(n)* durch die Verknüpfung der zugehörigen Matrizen finden:

$$ {}^0T_n = {}^0T_1 \cdot {}^1T_2 \cdot \ldots \cdot {}^{n-1}T_n = F(\underline{\theta}) $$

$$ {}^0T_n = \begin{bmatrix} n_n & s_n & a_n & p_n \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} {}^0R_n & {}^0p_n \\ 0 & 1 \end{bmatrix} \tag{5.4} $$

Hier sind $[n_n, s_n, a_n] = {}^0R_n$ die Orientierungsmatrix *(3x3)* des *n*- ten Koordinatensystems bezogen auf das Basiskoordiantessystem und ${}^0p_n = (x_n, y_n, z_n)$ der Positionsvektor *(3x1)* zum *n*- ten Koordinatensystemursprung. Wird auf das Koordinatensystem des Endeffektors Bezug genommen, so wird die Transformationsmatrix für den Roboter *amtec r7* mit sieben Gelenken weiter wie folgt spezifiziert:

$$ T = {}^0T_7 = {}^0T_1 \cdot {}^1T_2 \cdot \ldots \cdot {}^6T_7 = \begin{bmatrix} {}^0R_7 & {}^0p_7 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} n & s & a & p \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.5} $$

Tabelle 5.1 zeigt die geometrischen Beziehungen der einzelnen Teile des Roboters *amtec r7* zueinander durch DH-Parameter (θ_i, α_i, $\boldsymbol{a_i}$, $\boldsymbol{d_i}$), die durch oben beschriebene Regeln bestimmt wurden. Da bei realen Robotern die einzelnen Achsen parallel oder senkrecht zueinander angeordnet sind, treten gegenüber dem allgemeinen Fall wesentliche Vereinfachungen auf. Einige DH-Parameter werden dadurch zu Null, wodurch sich die späteren Berechnungen vereinfachen. Da die Achsen 1, 2, 3, 6 und 7 einen negativen Drehsinn haben, erhalten in diesen Fällen die Achswinkel θ_1, θ_2, θ_3, θ_6 und θ_7 ein negatives Vorzeichen. Die Achswinkel θ_5 und θ_7 sind entsprechend um +90° und -90° entlang der Z-Achse verdreht. Da es keine Verschiebungen entlang der x_i- Achse gibt, sind alle Größen a_i gleich Null.

Achse *i*	θ_i	α_i	d_i*(mm)*	a_i*(mm)*
1	$-\theta_1$	90°	335	0
2	$-\theta_2$	- 90°	0	0
3	$-\theta_3$	90°	300	0
4	θ_4	- 90°	0	0
5	θ_5 + 90°	90°	305	0
6	$-\theta_6$	- 90°	0	0
7	$-\theta_7$ - 90°	0°	200	0

Tabelle 5.1: Die DH-Parameter d_i, a_i, α_i und θ_i für den Roboter amtec r7

Setzt man die Parameter α_i, d_i und a_i aus der Tabelle 5.1 in die Gleichung (5.2), dann können die Transformationsmatrizen für den Roboter *amtec r7* aufgestellt werden:

$$
{}^0T_1 = \begin{bmatrix} \cos\theta_1 & 0 & -\sin\theta_1 & 0 \\ -\sin\theta_1 & 0 & \cos\theta_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad
{}^1T_2 = \begin{bmatrix} \cos\theta_2 & 0 & \sin\theta_2 & 0 \\ -\sin\theta_2 & 0 & \cos\theta_2 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^2T_3 = \begin{bmatrix} \cos\theta_3 & 0 & -\sin\theta_3 & 0 \\ -\sin\theta_3 & 0 & \cos\theta_3 & 0 \\ 0 & 1 & 0 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad
{}^3T_4 = \begin{bmatrix} \cos\theta_4 & 0 & -\sin\theta_4 & 0 \\ \sin\theta_4 & 0 & \cos\theta_4 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^4T_5 = \begin{bmatrix} -\sin\theta_5 & 0 & -\cos\theta_5 & 0 \\ \cos\theta_5 & 0 & -\sin\theta_5 & 0 \\ 0 & 1 & 0 & d_5 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad
{}^5T_6 = \begin{bmatrix} \cos\theta_6 & 0 & \sin\theta_6 & 0 \\ -\sin\theta_6 & 0 & \cos\theta_6 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

$$
{}^6T_7 = \begin{bmatrix} -\sin\theta_7 & \cos\theta_7 & 0 & 0 \\ -\cos\theta_7 & -\sin\theta_7 & 0 & 0 \\ 0 & 0 & 1 & d_7 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5.6}
$$

Die Position und Orientierung des Endeffektors ${}^0T_7$ lässt sich aus der Gl. (5.5) nach Multiplikation der sieben Matrizen (siehe Gl.5.6) darstellen.
Die Vektoren ***n***, ***s***, ***a*** und ***p*** in der Gleichung (5.5) sind die das Greiferkoordinatensystem beschreibenden Achsen (siehe Abb. 5.3). Die z_7- Achse ist der Annäherungsvektor ***a***, der senkrecht zur Handfläche gerichtet ist. Der y_7- Achse entspricht der Richtungsvektor ***s***, der in Richtung der Fingerbewegung beim Öffnen und Schließen zeigt. Die x_7- Achse ist der Normalvektor ***n***, der orthogonal zu den beiden anderen ist. Während ***a***, ***s*** und ***n*** die Orientierung des Greifers beschreiben, ist ***p*** der Richtungsvektor vom Ursprung des Basiskoordinatensystems zum Greiferkoordinatensytem.

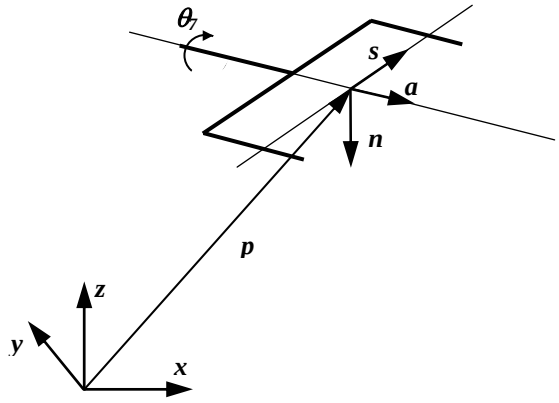


Abbildung 5.3: Koordinatensystem der Greiferhand

Die bisher verwendete Rotationsmatrix ***R*** bzw. die drei orthogonalen Vektoren ***a***, ***s*** und ***n*** sind nicht die einzige Methode, um Orientierungen zu beschreiben. Die Rotationsmatrix R enthält insgesamt 9 Parameter für 3 Freiheitsgrade (die Drehungen um x-, y- und z- Achsen). Zur Beschreibung von 3 Freiheitsgraden, d.h. von 3 voneinander unabhängigen

Bewegungsmöglichkeiten (Drehungen), reichen jedoch 3 Größen völlig aus. Für die Transformation einer Rotationsmatrix $\boldsymbol{R}$ in so genannte Eulersche Winkel mit 3 Parametern (γ, β, α), die die Beschreibung der Orientierung des Endeffektors vereinfachen, sind verschiedene Methoden geeignet. Sie basieren unter anderem darauf, dass es durch eine beliebige Sequenz von Drehungen möglich ist, jede beliebige Orientierung in Raum einzunehmen. Dabei lässt sich die Orientierung als Drehung eines Koordinatensystems gegenüber einem anderen Koordinatensystem beschreiben. Aus drei Rotationssequenzen [28] wurde „Rollen, Gieren, Nicken" als Sequenz gewählt. Bei dieser Methode werden die Winkel durch Rotationen um die Achsen des Referenzkoordinetensystems bestimmt (siehe Abb. 5.4).

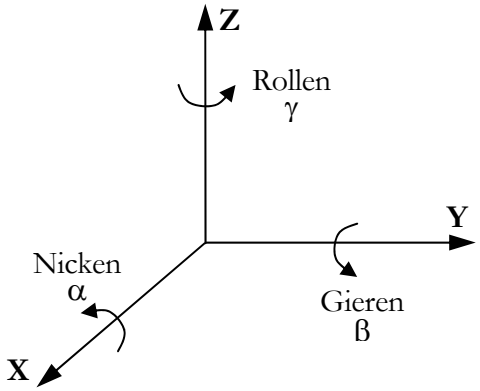


Abbildung 5.4: Drehungen eines Koordinatensystems um die Achsen X, Y und Z.

Hierbei wurde die Reihenfolge der einzelnen Rotationen wie folgt festgelegt:

1. Drehung um die z-Achse mit dem Winkel γ (Rotationsmatrix $R_{z\gamma}$)
2. Drehung um die neue y-Achse mit dem Winkel β (Rotationsmatrix $R_{y\beta}$)
3. Drehung um die neue x-Achse mit dem Winkel α (Rotationsmatrix $R_{x\alpha}$)

Die Gesamtrotationsmatrix ergibt sich nach [28] aus:

$$R_{zyx} = rot(z,\gamma)rot(y,\beta)rot(x,\alpha) \tag{5.7}$$

oder

$$R = R_{z\gamma}R_{y\beta}R_{x\alpha} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & os\beta \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}. \tag{5.8}$$

Damit gilt:

$$R = \begin{bmatrix} \cos\gamma\cos\beta & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\gamma\cos\beta & \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{5.9}$$

Es wird von der Beschreibung einer Rotationsmatrix durch Eulersche Winkel ausgegangen:

$${}^0R_n = R(\gamma,\beta,\alpha) = R_{z,\gamma}R_{y,\beta}R_{x,\alpha}, \tag{5.10}$$

Ziel der Rücktransformation ist hier die Bestimmung der Winkel γ, β und α aus Elementen von ${}^0R_n$ (siehe Gl. 5.4). Für die Orientierung des Endeffektors in der Winkelform gilt demnach:

$$\begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix} = \begin{bmatrix} \cos\gamma\cos\beta & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\gamma\cos\beta & \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \quad (5.11)$$

Hieraus kann man 9 Bestimmungsgleichungen ablesen

$$\begin{aligned} n_x &= \cos\gamma\cos\beta \\ n_y &= \sin\gamma\cos\beta \\ n_z &= -\sin\beta \\ s_x &= \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha \\ s_y &= \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha \\ s_z &= \cos\beta\sin\alpha \\ a_x &= \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ a_y &= \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ a_z &= \cos\beta\cos\alpha \end{aligned} \quad (5.12)$$

Aus (5.12) ergeben sich folgende Lösungsmöglichkeiten:

$$\begin{aligned} \beta &= -\arcsin(n_z) \\ \alpha &= \arcsin(s_z / \cos\beta) \\ \gamma &= \arcsin(n_y / \cos\beta) \end{aligned} \quad (5.13)$$

Diese Lösungsgleichungen sind jedoch unvollständig wegen der Verwendung der Arkussinusfunktion, die nur im Bereich von $-\pi/2$ bis $\pi/2$ definiert ist. Es ist aber notwendig, die Eulersche Winkel im Bereich [-π, π] zu definieren, um beliebige Orientierungen des Endeffektors zu beschreiben. Deshalb soll eine Beschreibung unter Verwendung der speziellen Arkustangensfunktion *arctan(y/x)*, die in der C-Programmbibliothek zur Verfügung steht, gefunden werden. Diese Funktion berechnet den Arkustangens aus den Parametern *x* und *y* im Bereich [-π, π].

$$\theta = \arctan\left(\frac{y}{x}\right) = \arctan\left(\frac{\sin\theta}{\cos\theta}\right) = \arctan\big(\tan(\theta)\big), \quad (5.14)$$

Dieser entspricht dem Arkustangens aus *x/y*. Allerdings muss das Vorzeichen beider Parameter der Arkustangensfunktion (5.14) ausgewertet werden und so der Quadrant des Ergebnisses bestimmt werden.

$$\theta = \arctan(y, x) = \begin{cases} 0^\circ \le \theta \le 90^\circ & \textit{für } +x \textit{ und } +y \\ 90^\circ \le \theta \le 180^\circ & \textit{für } -x \textit{ und } +y \\ -180^\circ \le \theta \le -90^\circ & \textit{für } -x \textit{ und } -y \\ -90^\circ \le \theta \le 0^\circ & \textit{für } +x \textit{ und } -y \end{cases} \quad (5.15)$$

Hierfür wird die Gleichung (5.10) umgeformt:

$$R_{z,\gamma}^{-1} \cdot {}^{0}R_{n} = R_{y,\beta} \cdot R_{x,\alpha} \tag{5.16}$$

Aufgrund von (5.16) in Verbindung mit (5.8) gilt dann:

$$\begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} n_x & s_x & a_x \\ n_y & s_y & a_y \\ n_z & s_z & a_z \end{bmatrix} = \begin{bmatrix} \cos\beta & \sin\beta\sin\alpha & \sin\beta\cos\alpha \\ 0 & \cos\alpha & -\sin\alpha \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{5.17}$$

bzw.:

$$\begin{bmatrix} n_x\cos\gamma + n_y\sin\gamma & s_x\cos\gamma + s_y\sin\gamma & a_x\cos\gamma + a_y\sin\gamma \\ n_y\cos\gamma - n_x\sin\gamma & s_y\cos\gamma - s_x\sin\gamma & a_y\cos\gamma - a_x\sin\gamma \\ n_z & s_z & a_z \end{bmatrix} = \\ = \begin{bmatrix} \cos\beta & \sin\beta\sin\alpha & \sin\beta\cos\alpha \\ 0 & \cos\alpha & -\sin\alpha \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{5.18}$$

Nimmt man die ersten Elemente der zweiten Zeile beider Matrizen in Gleichung (5.18), so gilt:

$$n_y\cos\gamma - n_x\sin\gamma = 0 \tag{5.19}$$

und daraus erhält man

$$\tan\gamma = \frac{\sin\gamma}{\cos\gamma} = \frac{n_y}{n_x} \tag{5.20}$$

und

$$\gamma = \arctan(n_y, n_x) \tag{5.21}$$

Es existiert jedoch eine Doppeldeutigkeit, wenn $\gamma = \arctan(-n_y, -n_x)$ ist. Daher wird zusätzlich das Vorzeichen der Argumente nach Gl. (5.15) ausgewertet, um den Eulerschen Winkel im Bereich [-π, π] zu bestimmen. Für den Fall, dass n_x und n_y gleich Null sind, ist γ nicht definiert. In diesem Fall wird der Winkel γ Null gesetzt.
Wenn γ ermittelt ist, sind alle Elemente der linken Seite der Gleichung (5.18) bestimmt, da die Parameter n_i , s_i und a_i nach der Multiplikation aller Matrizen der Gleichung (5.6) für den 7-achsigen Roboter *amtec r7* bekannt sind. Nimmt man die ersten Elemente der ersten und der dritten Zeilen beider Matrizen in der Gleichung (5.18), so kann man auch folgendes Gleichungssystem ablesen:

$$\begin{cases} \cos\beta = n_x\cos\gamma + n_y\sin\gamma \\ \sin\beta = -n_z \end{cases} \tag{5.22}$$

Nimmt man die zweiten und der dritten Elemente der zweiten Zeile beider Matrizen in der Gleichung (5.18), so gilt:

$$\begin{cases} \cos\alpha = -s_x\sin\gamma + s_y\cos\gamma \\ -\sin\alpha = -a_x\sin\gamma + a_y\cos\gamma \end{cases} \tag{5.23}$$

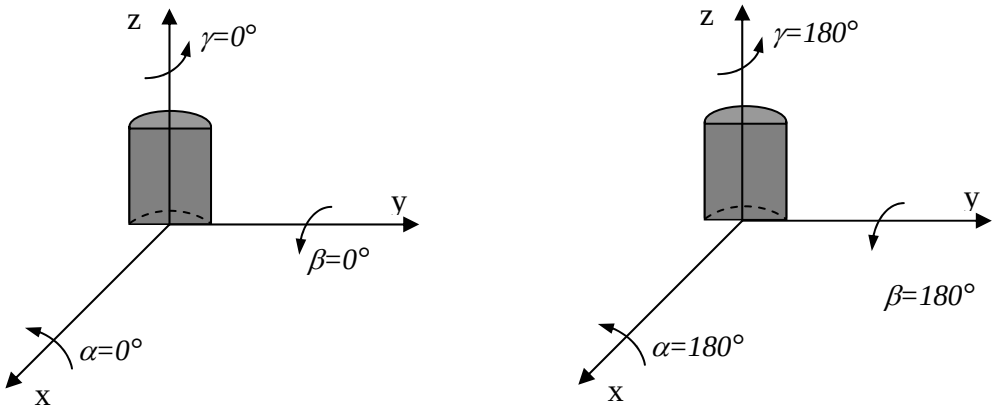
Aus der Gl. (5.22) folgt:

$$\beta = \arctan\left(\frac{\sin\beta}{\cos\beta}\right) = \arctan(-n_z,\quad n_x\cos\gamma + n_y\sin\gamma) \tag{5.24}$$

und aus der Gl. (5.23) ergibt sich:

$$\alpha = \arctan\left(\frac{\sin\alpha}{\cos\alpha}\right) = \arctan(a_x\sin\gamma - a_y\cos\gamma,\quad -s_x\sin\gamma + s_y\cos\gamma) \tag{5.25}$$

Die Vorzeichen der Argumente *x* und *y* in der Gleichungen (5.24) und (5.25) müssen auch nach der Gl. (5.15) ausgewertet werden. Somit sind alle drei Winkel γ, β und α bestimmt worden, so dass das Koordinatensystem hinsichtlich der Orientierung rücktransformiert werden kann. Damit ist eine Vorschrift vorhanden, wie die Winkel γ, β und α verändert werden müssen, um die gewünschte direkte Transformation zu realisieren.



1) α =β=γ=0° 2) α =β=γ=180°

Abbildung 5.5: Zwei Arten der Repräsentation von ein und derselben Orientierung.

Ein und dieselbe Orientierung des Endeffektors kann in zwei verschiedenen Arten beschrieben werden (Abb. 5.5):

1. Wenn die Winkel α, β und γ gleich Null sind.
2. Wenn die Winkel α, β und γ um 180° gedreht sind.

Bei der Lösung der Gleichungen (5.21), (5.24) und (5.25) wurde die Orientierung nach der 1. Art bestimmt, wenn die Winkel α, β und γ kleiner als *90°* sind. Wenn einer der drei Winkel größer als 90° ist, dann wurde die Orientierung nach der 2.Art berechnet. Um die Orientierung in passender Form (1.Art) zu repräsentieren, wurden bei $|\beta| \geq 90°$ alle drei Winkel um 180° korrigiert.
Unter Voraussetzung, dass die genauen geometrischen Größen der Gelenkwinkel des Roboters bekannt sind, kann somit nun die genaue Position und Orientierung des Endeffektors mittels der direkten Kinematik erfolgreich ermittelt werden.

5.1.2 Simulative und experimentelle Untersuchungen

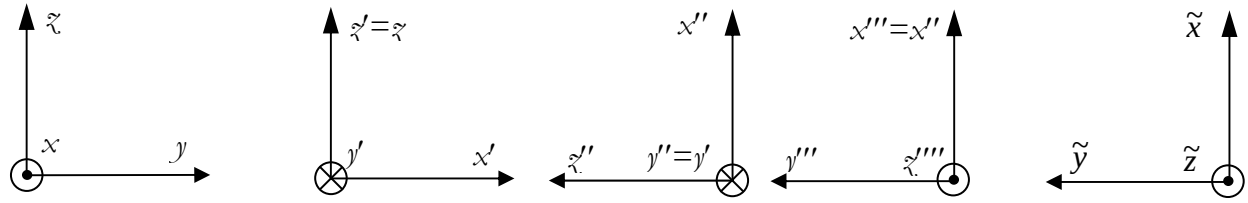
Gemäß der im Abschnitt 5.1.1 beschriebenen Vorgehensweise zur Berechnung der direkten Kinematik wurde ein C-Programm „dirkin7.c“ geschrieben, das aus den 7 Gelenkwinkeln des Roboters die Position (P_x, P_y, P_z) und die Orientierung in Eulerschen Winkel (α, β, γ) des

Endeffektors berechnet. Um zu ermitteln, ob die DH-Parameter richtig bestimmt wurden und das Programm fehlerfrei arbeitet, wurden simulative und experimentelle Untersuchungen für verschiedenen Gelenkwinkel des Roboters durchgeführt und miteinander verglichen. Die Ergebnisse sind in Tabelle 5.2 zusammengestellt. Einerseits wurden dabei also die Position und Orientierung des Endeffektors mittels simulativen Untersuchungen ermittelt, anderseits wurden die Positionen direkt am Roboter vermessen und die Orientierung als Dreibein des Endeffektors $(\tilde{z}, \tilde{y}, \tilde{x})$ aufgezeichnet. Dabei wurden 8 verschiedene Konfigurationen des Roboters betrachtet. Für die deutliche grafische Darstellung der in der Tabelle 5.2 erfassten Ergebnisse wurden die Gelenke jeweils um 90° gedreht. Die resultierenden Eulerwinkel beschreiben die Orientierung bezüglich des Basiskoordinatensystems. Deshalb muss man für die Prüfung der Richtigkeit der simulativen Berechnungen der Orientierung des Endeffektors folgende Reihenfolge der Drehungen des Basiskoordinatensystems vornehmen:

1. Drehung um die z-Achse mit dem Winkel γ
2. Drehung um die neue y-Achse (y') mit dem Winkel β
3. Drehung um die neue x-Achse (x'') mit dem Winkel α

Danach muss das Dreibein des Endeffektors mit dem auf solche Weise gedrehten Basiskoordinatensystem übereinstimmen. Da die einzelne Gelenke des Roboters in unseren Untersuchungen jeweils um 90° gedreht werden, betrachten wir als Beispiel die Drehungen des Basiskoordinatensystems mit den Winkel γ, β und α um jeweils 90°.

Wie man aus der Abbildung 5.6 sieht, stimmt das Basiskoordinatensystem (a) nach der Reihe von Drehungen um γ, β und α (b), die analytisch durchgeführt wurden, mit dem Dreibein des Endeffektors (c) überein, wenn die Eulerschen Winkel richtig gefunden wurden. Es ist auch nötig zu bemerken, dass die positive Richtung der Drehung um eine Achse mit der Richtung der Drehung der rechten Hand übereinstimmt.



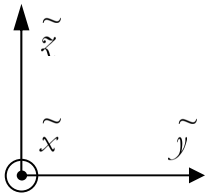
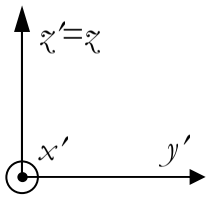
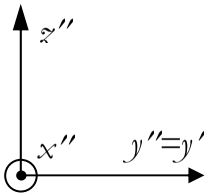
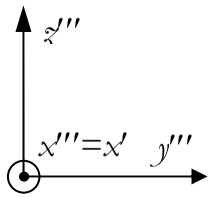
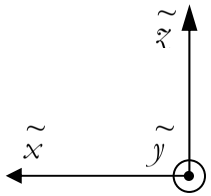
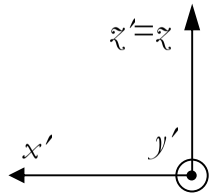
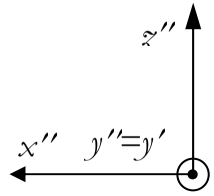
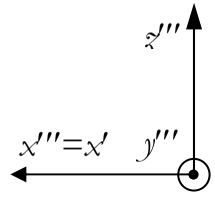
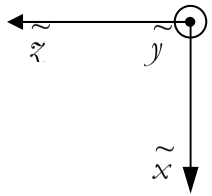
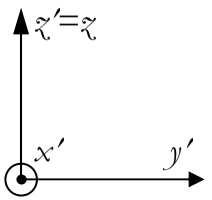
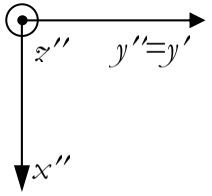
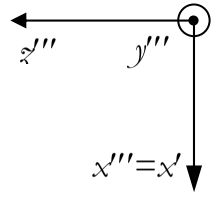
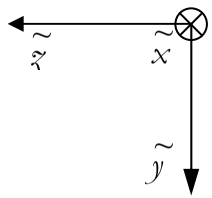
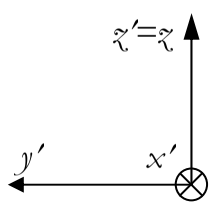
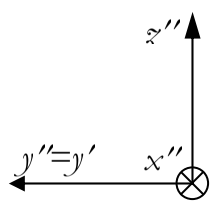
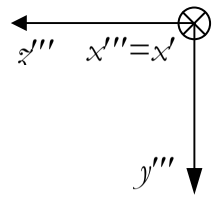
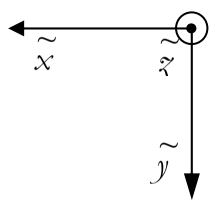
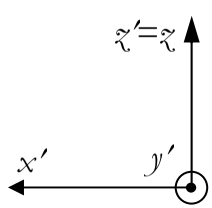
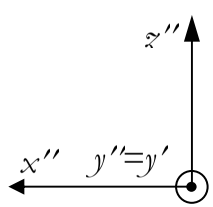
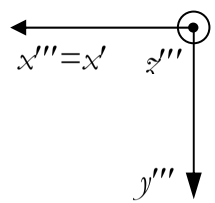
a) Basiskoordinaten-system

b) Drehungen um γ, β und α entsprechend der Reihenfolge

c) Dreibein des Endeffektors

Abbildung 5.6: Verifikation der Übereinstimmung des Endeffektordreibeins nach Drehung des Koordinatensystems um γ, β und α

Die Symbole ⊗ und ⊙ in der Abbildung 5.6 zeigen die jeweiligen Richtungen des Pfeils einer Koordinatenachse. Das Symbol ⊙ zeigt, dass die Spitze des Pfeils aus der Ebene heraus gerichtet ist. Dagegen zeigt das Symbol ⊗, dass die Spitze des Pfeils in die Ebene hinein zeigt.

№	Drehwinkel, Grad							Position, mm			Orientierung, Grad			Experiment	Verifikation		
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	P_x	P_y	P_z	α	β	γ	Ergebnisse	Drehung γ (um Achse z)	Drehung β (um Achse y')	Drehung α (um Achse x'')
1	0	0	0	0	0	0	0	0	0	1140	0	0	0				
2	90	0	0	0	0	0	0	0	0	1140	0	0	-90				
3	90	90	0	0	0	0	0	0	-805	355	90	90	0				
4	90	90	90	0	0	0	0	0	-805	335	-90	0	180				
5	90	90	90	90	0	0	0	505	-300	335	-90	0	-90				

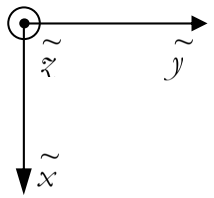
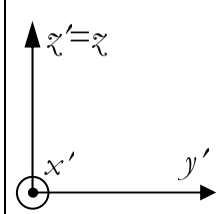
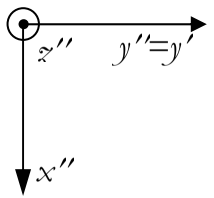
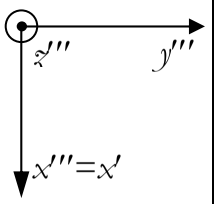
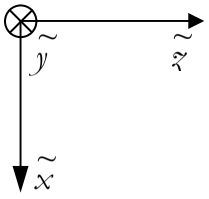
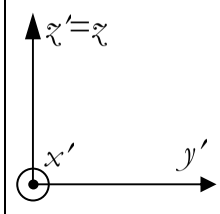
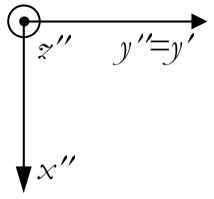
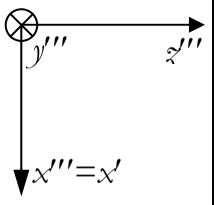
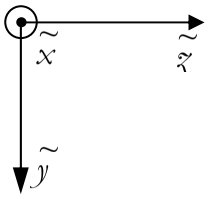
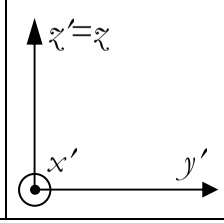
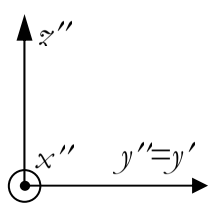
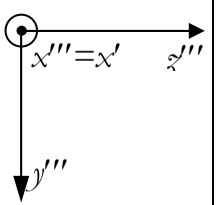
№	Drehwinkel, Grad							Position, mm			Orientierung, Grad			Experiment	Verifikation		
	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	θ_7	P_x	P_y	P_z	α	β	γ	Ergebnisse	Drehung γ (um Achse z)	Drehung β (um Achse y')	Drehung α (um Achse x'')
6	90	90	90	90	90	0	0	505	-300	335	0	90	0				
7	90	90	90	90	90	90	0	305	-100	335	-90	90	0				
8	90	90	90	90	90	90	90	305	-100	335	-90	0	0				

Tabelle 5.2: Ergebnisse der simulativen und experimentellen Untersuchung der direkten Kinematik des Roboters „amtec r7“

5.2 Inverse Kinematik

5.2.1 Grundlagen

Die Aufgabe der inversen Kinematik ist es, aus einer vorgegebenen Position $P=(x, y, z)$ und Orientierung R bzw. (α, β, γ) des Endeffektors die entsprechenden Gelenkwinkel θ_i der Einzelachsen des Roboters zu bestimmen, die diese realisieren. Dabei ist die inverse Kinematik mathematisch wesentlich komplexer als die direkte Kinematik. Für den 7-achsigen Roboter *amtec r7* lässt sich die Gleichung zur Bestimmung der inversen Kinematik zunächst wie folgt spezifizieren

$$\theta_i = f^{-1}({}^0T_7) = f^{-1}\begin{pmatrix} {}^0R_7 & {}^0P_7 \\ 0 & 1 \end{pmatrix} = f^{-1}(x, y, z, \alpha, \beta, \gamma), \tag{5.26}$$

wobei $\theta_i=(\theta_1, \theta_2, \ldots, \theta_7)$ die sieben Gelenkwinkel des Roboters sind.
Es existieren folgende erschwerende Bedingungen für die Lösung der inversen Kinematik:

- **Nichtlinearität**
  Für die inverse Transformation existieren keine allgemeinen Lösungsmethoden auf Grund ihrer starken Nichtlinearität und Abhängigkeit von jeweiliger Hardwarestruktur des Roboters. Die Bestimmung ist weitaus komplizierter als im direkten Fall, da kein alles umfassender analytischer Algorithmus existiert, durch den zu jeder beliebigen Kinematik die dazugehörige explizite Rücktransformation bestimmt werden kann. Diese Schwierigkeit ist mathematisch damit verbunden, dass für einen Satz von nichtlinearen trigonometrischen Gleichungen geeignete Bestimmungsgleichungen zur inversen Kinematik gefunden werden müssen.

- **Mehrdeutigkeit**
  Außerdem liefert die Rücktransformation der Gelenkwinkel nach Gl. 5.26 nicht immer eindeutige Lösungen. Das heißt, ein und dieselbe Lage des Endeffektors kann häufig durch verschiedene Stellungen des Roboters erreicht werden (siehe Abb. 5.7). Dabei unterscheidet man
  - endliche Mehrdeutigkeit bei nichtredundanter Kinematik und
  - unendliche Mehrdeutigkeit bei redundanter Kinematik.

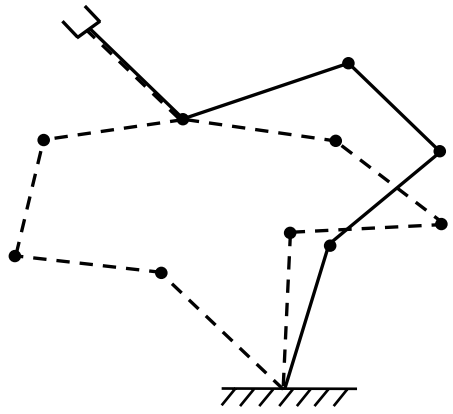


Abbildung 5.7: Mehrdeutigkeit der inversen Kinematik eines Roboters

Die zweckmäßige Konfiguration aus diesen möglichen Stellungen auszuwählen, ist wiederum eine wichtige Aufgabe bei der Lösung des Problems der inversen Kinematik.

- **Singularität spezieller Punkte**
  Ein weiteres Problem stellen die so genannten singulären Punkte des Roboters dar. Das sind Stellungen, in denen der Roboter im kartesischen Raum einen oder mehrere Freiheitsgrade verliert. Solche Punkte treten z.B. immer an der Grenze des Arbeitsraumes auf. Dies bedeutet, dass sich der Arm in einer Raumrichtung oder Drehrichtung nicht bewegen kann. Versucht man dies trotzdem, so sind große Gelenkgeschwindigkeiten notwendig, um nur eine kleine Bewegung in diese Richtung zu erreichen. Ein redundanter Roboter besitzt die Möglichkeit mit seinen zusätzlichen Freiheitsgraden viele singuläre Punkte zu umgehen.

- **Rechenzeitforderungen**
  Da die Berechnung der inversen Kinematik im *Online*-Betrieb des Roboters notwendig ist, ergeben sich darüber hinaus besondere Anforderungen an die Schnelligkeit ihrer Berechnung. Die Minimierung dieser Rechenzeit ist von wesentlicher Bedeutung, weil diese Zeit bei der Verarbeitung von Sensorsignalen im Kameramodell (siehe Abb. 2.3) als reine Totzeit eingeht. Daher gibt es große Bemühungen, einmal eine allgemeine, für beliebige Roboter gültige und zum anderen numerisch effiziente Möglichkeit für die Bestimmung der inversen Kinematik zu finden.

Aktuell existieren folgende Verfahren zur Bestimmung der inversen Kinematik in der Literatur:

1. **Analytische Methoden**
   Eine analytische Lösung mit vertretbarem Aufwand kann im Allgemeinen nur für Spezialfälle berechnet werden. Aus diesem Grund sind die meisten Industrieroboter kinematisch einfach gebaut, d.h. die Bewegungsachsen verlaufen parallel oder schneiden sich senkrecht. Für sie existieren damit auch geschlossene Lösungen. Der Nachteil der analytischen Lösungen liegt allerdings darin, dass sie häufig zu mehreren Sätzen von Achskoordinatenvektoren führt, die alle der gleichen Position und Orientierung des Endeffektors im Raum entsprechen. Auf Grund dieser Mehrfachlösungen ist ein Entscheidungsprozess zur Auswahl eines geeigneten Achskoordinatenvektors erforderlich. Für den Roboter mit mehr als zwei parallelen Achsen ist dieses Verfahren nicht mehr anwendbar. Außerdem versagt diese Methode wie in unserem Fall bei den redundanten Manipulatoren, wo die Anzahl der Gleichungen kleiner ist, als Anzahl der Unbekannten (*unterbestimmtes Gleichungssystem*). Die analytische Methoden wurden in [28], [70] ausführlich betrachtet.

2. **Geometrische Methoden**
   Auf Grund der vorliegenden Kinematik und Vorgabe der Lage des Endeffektors, kann mit Hilfe einer Skizze geometrisch das Problem der inversen Kinematik gelöst werden. Dann werden anhand dieser Skizze die trigonometrischen Beziehungen der Achsen zueinander aufgestellt. Diese Funktionen werden bzgl. der einzelnen Gelenkwinkel des Roboters gelöst.
   Die geometrischen Methoden führen zum Ziel, sind aber
   - recht aufwändig abzuleiten,
   - nicht übertragbar auf die andere Kinematiken und
   - führen bei komplexen Kinematiken zu umfangreichen „Kunstwerken“.

- Bei mehrdeutigen Lösungen ist die Auswahl der günstigsten Achsenkonfiguration erforderlich.

Weitergehende und tiefere Betrachtungen dazu erfolgen in [28] und [70].

3. **Optimierungsmethoden**
   Die Optimierungsmethoden gehören derzeit zu den universellen Methoden der Bestimmung der inversen Kinematik. Sie wurden bereit in den 80iger Jahren zu diesem Zwecke eingesetzt, allerdings war der Rechenaufwand für die damalige Rechentechnik zu groß, um die inverse Kinematik in Echtzeit für mehrachsige Roboter zu realisieren. Zur Zeit sind die Optimierungsmethoden zur Lösung dieses Problems meistens für die redundanten Roboter einsetzbar.
   Zur Auflösung der Redundanz von Robotern wurden in den letzten Jahren viele Methoden und Vorgehensweisen entwickelt. Man unterscheidet Methoden, die lokal optimale, und Methoden, die global optimale Lösungen suchen. Die zweite Klasse von Methoden versucht für den Weg des Endeffektors einen optimalen Pfad zu finden. Die lokalen Methoden wählen bei jedem Teilschritt unabhängig die lokal optimale Lösung. Globale Methoden (zum Beispiel über Pontrjagins Maximumprinzip, siehe [11] oder [101]) sind komplex und äußerst rechenaufwändig. Die lokalen Methoden bieten jedoch keine Garantie für global optimale Roboterbewegungen, sind jedoch Teile der globalen Lösungen. Im Rahmen dieser Arbeit wurden nur lokale Methoden näher untersucht, um die schnellere Berechnung der inversen Kinematik in Echtzeit zu sichern. Nenchev gibt in [66] eine Übersicht über die Möglichkeiten, lokal optimale Lösungen zu suchen. Bereits dieser frühe Artikel gibt einen guten Überblick über die unterschiedlichen Möglichkeiten und Ideen. Gemeinsam ist allen Ansätzen, dass versucht wird, ein unterbestimmtes Gleichungssystem durch zusätzliche Gleichungen zweckmäßig zu erweitern. Diese Gleichungen haben die Aufgabe, die zusätzlichen Freiheitsgrade des Roboters sinnvoll zu nutzen. Zu den lokalen Methoden gehören Z. B. die Methode der *task priorities* (TP) [101] und die Lösung mit Hilfe der *extended Jacobian method* EJM [89]. Die Methode der TP besitzt eine anschauliche physikalische Deutung als Regelungsalgorithmus im Geschwindigkeitsbereich. Innerhalb eines Iterationsschrittes berechnet man mögliche Gelenkänderungen, die die gewünschte Geschwindigkeit des Endeffektors ergeben. Die EJM- Methode liefert nur Aussagen über Extrema und nicht darüber, ob es sich um ein Maximum oder ein Minimum handelt. Park, Chung und Youm [69] stellen eine Methode zur Verfügung, mit deren Hilfe diese Frage geklärt wird. Die Berechnung ist aber sehr aufwändig und geht davon aus, dass bereits ein Startwert vorliegt, der das Problem löst.

4. **Neuronale Netze**
   Neuronale Netze wurden bereits mehrfach zur Bestimmung der inversen Kinematik von Roboterarmen eingesetzt. Die Motivation der Autoren zur Beschäftigung mit dieser Fragestellung war dabei immer die Bewältigung der gestellten Aufgabe durch das Erlernen an Hand von Beispielen an Stelle des Einsatzes algorithmischer Lösungen. Erste Versuche zum Einsatz neuronaler Netze bei der inversen Kinematik [32] suchten nach globalen Lösungen, die für jeden Punkt des Arbeitsraumes die korrespondierende Konfiguration approximieren. Dabei treten Fehler von bis zu zehn Zentimetern auf. Mit zusätzlichen Newton- Raphson Iterationen zur Bestimmung der exakten Konfiguration und Nutzung der Netzwerklösung als deren Startwert [33] wurde dieser

Ansatz verbessert. Ein ähnlicher Ansatz, bei dem Kohonen Merkmalskarten die inverse Kinematik eines Dreigelenkarmes mit stückweise konstanten Funktionen global memorieren, wurde bereits von Brause [10] vorgeschlagen. Dabei stand vor allem die Menge an benötigtem Speicher für eine ausreichende Genauigkeit (≤5%) im Vordergrund. Neuronale Netze werden auch für die inverse Kinematik bei der visuo-motorischen Regelung eingesetzt [94]. Dabei wurde eine globale inverse Abbildung gelernt. Das eingesetzte Netzwerk ist eine selbstorganisierende Karte, die aber nicht überwacht lernt, sondern den Fehler im Arbeitsraum dazu benutzt, die Netzwerkparameter zu adaptieren. Für die differentielle inverse Kinematik (siehe Gl. (5.80)) besteht der Eingangsvektor aus den Gelenkwinkeln und der differentiellen Wunschbewegung in kartesischen Koordinaten. Der Ausgangsvektor besteht aus differentiellen Gelenkwinkeln und Wunschbewegungen, die linear sind, wobei die Koeffizienten von den aktuellen Gelenkwinkeln abhängen [99]. Deshalb wird der Eingangsvektor in einen linearen und nichtlinearen Teil getrennt, die dann unterschiedlich behandelt werden. Der nichtlineare Teil ist der Eingang eines Netzes, dessen Ausgang die Koeffizienten der linearen Beziehung zwischen differentiellen Wunschbewegungen und Gelenkwinkeln ist. Diese Vorgehensweise wurde von [100] für die Verwendung bei der inversen Kinematik vorgeschlagen, wobei das eingesetzte neuronale Netz alternativ ein Multilayer Perceptron oder ein RBF Netz war. Es wurden ein Dreigelenkroboterarm in drei Dimensionen des Arbeitsraums und ein Sechsgelenkroboterarm behandelt, wobei keine absoluten Werte des Fehlers in kartesischen Koordinaten und kein Arbeitsraum angegeben wurden. Mit einem ähnlichen Vorgehen [29] wurde die inverse Jacobimatrix (siehe im Abschnitt 5.2.3) mit vier Komponenten für einen Zwei- Gelenk Roboterarm aus 81 Beispielen gelernt, wobei Singularitäten explizit ausgeschlossen wurden. Ein mittlerer Fehler von 1-3% wurde nach dem Training gemessen. In [37] wurde ein Verfahren zum Training einer inversen Jacobimatrix für einen sechsachsigen Roboter entwickelt. Dabei wurden die Singularitätspunkte einbezogen. Die Ergebnisse des Trainings erwiesen sich ausreichend, sind aber nur für einen kleinen Arbeitsraum (40x40x40 cm) des Roboters gültig.

Insgesamt lässt sich sagen, dass die Bestimmung der inversen Kinematik die Verwendung großen Mengen von Daten und sehr großen Rechenaufwand erfordert. Für eine beschränkte Anzahl von Gelenken des Roboters oder kleine Arbeitsräume sind Neuronale Netze zur Lösung des Problems der inversen Kinematik einsetzbar. Wenn die inverse Kinematik eines redundanten Roboters für seinen gesamten Arbeitsraum bestimmt werden muss (wie in unserem Fall), sind Neuronale Netze aus den oben erwähnten Gründen nicht zweckmäßig. Die genaue Beschreibung des dabei zu lösenden Problems wird im Abschnitt 5.2.3 dargestellt.

Für diese Arbeit wurden deshalb zur Berechnung der inversen kinematischen Transformation zwei geeignete Methoden ausgewählt. Eine davon verwendet ein spezielles nichtlineares Optimierungsverfahren zur Berechnung der inversen Kinematik. Die andere ermittelt die inverse Kinematik mit Hilfe eines Neuronalen Netzes. Welches Optimierungsverfahren für die erste Methode am besten geeignet ist, wird im nächsten Abschnitt des Kapitels betrachtet. An ein Optimierungsverfahren, das innerhalb einer Steuerung verwendet werden soll, müssen neben der Forderung, eine Lösung für die Optimierungsaufgabe zu finden, folgende Forderungen gestellt werden:

1. Das Verfahren muss gute Konvergenz haben und stabil sein, da innerhalb jedes Steuertaktes eine Lösung gefunden werden muss.
2. Eine hohe Anzahl von Restriktionen (Nebenbedienungen) muss beachtet werden können, da jedes Gelenk eines Roboters bei der Bewegung Einschränkungen bezüglich der Bewegungsfreiheit unterworfen ist.
3. Da die Optimierung in jedem Steuertakt erneut aufgerufen wird, darf der Algorithmus nicht zu aufwendig werden, da sonst die innerhalb eines Steuertaktes zur Verfügung zu stellende Rechenzeit nicht realisierbar ist.
4. Das Verfahren muss eine ausreichende Robustheit besitzen, um die in der Praxis auftretenden Parameterfehler zuzulassen.

5.2.2 Grundlagen der nichtlinearen Optimierungsverfahren

Es gibt eine große Zahl an Optimierungsverfahren für die verschiedensten Anwendungen. Dabei sind die einzelnen Optimierungsverfahren jeweils nur für bestimmten Arten oder Strukturen von Fehlerfunktion (Zielfunktion) geeignet. Es zeigte sich, dass es nicht *den* Algorithmus gibt, der jedes Optimierungsproblem besser löst als andere, sondern dass ein günstiger Algorithmus jeweils nur in Bezug auf das Problem, d.h. in erster Linie auf die zu minimierende Fehlerfunktion, angegeben werden kann. Daraus folgt, dass unterschiedliche Optimierungsprobleme unterschiedliche Algorithmen zur Berechnung benötigen. Dabei kann man die Optimierungsverfahren folgendermaßen unterteilen:

- **Beschränkte und unbeschränkte Optimierung**
  Bei einem beschränkten Optimierungsproblem sind den Variablen Grenzen gesetzt. Zum Beispiel dürfen die Variablen nur positive Werte annehmen.
- **Globale und lokale Optimierung**
  Bei einer globalen Optimierung wird das absolute Minimum bzw. Maximum der Fehlerfunktion berechnet. Im Gegensatz dazu wird bei einer lokalen Optimierung ein Minimum/Maximum in einem Intervall berechnet.
- **Mit und ohne Verwendung der Ableitung**
  Bei bestimmten Arten der Optimierung werden die Ableitungen zur Lösungsfindung hinzugezogen.
- **Eindimensionale und mehrdimensionale Optimierung**
  Im eindimensionalen Fall ist die Berechnung auf eine eindimensionale Funktion mit nur einer Variablen begrenzt. Bei Berechnung einer mehrdimensionalen Funktion ist die Bildung der einzelnen Gradienten (z.B. Gradientverfahren) zu berücksichtigen.

Man entwickelt numerische Verfahren, um oben erwähnten Optimierungsprobleme möglichst schnell und leicht lösen zu können. Die numerischen Verfahren bei nichtlinearen Optimierungsproblemen können in der Regel nur ein lokales Minimum berechnen. Nichtlineare Abhängigkeiten liefern jedoch oft Zielfunktionen, die mehrere lokale Minima besitzen. Welches lokale Minimum berechnet wird, hängt dabei vom Startpunkt ab. Um einen geeigneten Startpunkt zu erhalten, muss man oft „experimentieren“. Außerdem sollte man immer überprüfen, ob das berechnete lokale Minimum tatsächlich eine brauchbare Lösung ist. In dieser Arbeit ist das nichtlineare Optimierungsproblem der direkten Kinematik eines redundanten Roboters zu behandeln. Ausgangspunkt hierfür ist folgende Gleichung:

$$(p_x, p_y, p_z, \alpha, \beta, \gamma)^T = f(\theta_n), \tag{5.27}$$

wobei (p_x, p_y, p_z) und (α, β, γ) Position und Orientierung des Endeffektors des Roboters sind. $f(\theta_n)$ ist eine mehrdimensionale trigonometrische Funktion, die von den einzelnen Gelenkwinkeln des Roboters abhängig ist, entspricht der analytisch lösbaren direkten Kinematik. Als Fehlerfunktion (Differenz zwischen der durch die Trajektorie vorgegebenen und der aus der Konfiguration des Roboters θ_n mittels direkter Kinematik berechneten Lage des Endeffektors) ergibt sich aus Gl. (5.27):

$$E(\theta_n) = \sum_{i=0}^{6} \left((p_x, p_y, p_z, \alpha, \beta, \gamma) - f_i(\theta_n) \right), \tag{5.28}$$

oder

$$\begin{aligned}
E_1(\theta_n) &= \left(p_x - f_1(\theta_n)\right), \\
E_2(\theta_n) &= \left(p_y - f_2(\theta_n)\right), \\
E_3(\theta_n) &= \left(p_z - f_3(\theta_n)\right), \\
E_4(\theta_n) &= \left(\alpha - f_4(\theta_n)\right), \\
E_5(\theta_n) &= \left(\beta - f_5(\theta_n)\right), \\
E_6(\theta_n) &= \left(\gamma - f_6(\theta_n)\right), \\
E(\theta_n) &= \sum_{i=1}^{6} E_i(\theta_n).
\end{aligned} \tag{5.29}$$

Diese Fehlerfunktion muss dabei minimiert werden, so dass für sämtliche Punkte im Arbeitsraum die entsprechende Konfiguration des Roboters (Vektor der Gelenkwinkel) gefunden wird. Da die Minimierung der Fehlerfunktion (5.28) in jedem räumlich kleinen Schritt des Steuertaktes entlang einer vorgegebenen Trajektorie durchgeführt werden soll, werden sich die ermittelten Gelenkwinkel von den im vorhergehenden Takt ermittelten nicht wesentlich unterscheiden. Dabei ist außerdem der Startpunkt für das Optimierungsverfahren bekannt, weil die den gemessenen Gelenkwinkeln entsprechende Position und Orientierung des Endeffektors (des Startpunktes) aus der direkten Kinematik bekannt ist. Um die Zielfunktion (5.28) zu minimieren, braucht man deshalb ein mehrdimensionales Optimierungsverfahren, das ein lokales Minimum dieser Funktion finden soll. Das stellt ein beschränktes Optimierungsproblem dar, da jeder Gelenkwinkel des Roboters gegebene Grenzbereiche der Bewegung einhalten muss.

Für die Minimierung der Fehlerfunktion (5.28) sind nach [50] am besten die Abstiegsmethoden geeignet. Die Idee der Abstiegsmethoden besteht in folgendem:

Um eine Funktion f mit der nichtleeren offenen Teilmenge V des gesamten Raums R^m zu minimieren, muss f in jedem Punkt $x \in V$ alle partiellen Ableitungen $f_{x_i}(x)$, $i=1,...,m$ besitzen. Ausgehend von einem Startwert $x_0 \in V$ erzeugen die lokalen Optimierungsverfahren eine Punktfolge $x_1, x_2, x_3, ..., x_k$, die gegen ein lokales Minimum $\hat{x}$ von $f(x)$ konvergiert.

Ein Startpunkt $x_0 \in V$ wird derart vorgegeben, dass

$$grad\ f(x_0) \neq O_m \tag{5.30}$$

ist, wobei O_m ein Nullvektor mit m-Elementen ist. Dann ist x_0 kein lokaler Minimalpunkt und damit auch kein Minimalpunkt von f auf V.
Man führt irgendeinen Vektor $h \in R^m$ (z.B. $h = -\,grad\ f(x_0)$) in der Gl. (5.30) derart ein, dass gilt:

$$h^T\ grad\ f(x_0) < 0\,. \tag{5.31}$$

Da die Teilmenge V offen ist, gibt es ein Koeffizient $\lambda_h > 0$ derart, dass gilt

$$x_0 + \lambda h \in V \ \text{ für alle } \ \lambda \in [0, \lambda_h], \tag{5.32}$$

und für genügend kleines $\lambda \in [0, \lambda_h]$ gilt

$$f(x_0 + \lambda h) < f(x_0)\,. \tag{5.33}$$

Man nennt daher jeden Vektor $h \in IR^m$ mit (5.31) *eine Abstiegsrichtung* von f in $x_0 \in V$. Den Vektor $h = -\,grad\ f(x_0)$ nennt man (im Falle (5.30)) *die Richtung des steilsten Abstieges* von f in $x_0 \in V$. Hat man ein $h \in IR^m$ mit (5.31) ermittelt und ein $\lambda_h > 0$ mit (5.32) gefunden, so bestimmt man ein $\lambda_0 \in [0, \lambda_h]$ mit

$$f(x_0 + \lambda_0 h) \le f(x_0 + \lambda h) \ \text{ für alle } \ \lambda \in [0, \lambda_h]. \tag{5.34}$$

Ein solches $\lambda_0 \in [0, \lambda_h]$ gibt es, da die Funktion $F(x) = f(x_0 + \lambda h)$ auf $[0, \lambda_h]$ stetig ist. Auf Grund von (5.33) ist $\lambda_0 \in [0, \lambda_h]$ und

$$f(x_1) = f(x_0 + \lambda_0 h) < f(x_0) \tag{5.35}$$

Damit ist bereits ein Schritt *des allgemeinen Abstiegsverfahrens* beschrieben. Dieser besteht also (unter Voraussetzung (5.30)) aus der Bestimmung einer Abstiegsrichtung $h \in IR^m$ mit (5.31) und einer *Schrittweite* $\lambda_h > 0$ mit $x_0 + \lambda_0 h \in V$ und (5.35), die z.B. durch Lösung der Minimierungsaufgabe (5.34) gewonnen werden kann. Ist $grad\ f(x_0 + \lambda_0 h) \neq O_m$, so kann der Schritt wiederholt werden.
Dabei erhält man die Iterationsformel für das allgemeinen Abstiegsverfahren:

$$x_{k+1} = x_k + \lambda_k h_k \tag{5.36}$$

Die Abb. 5.8 zeigt die möglichen Abstiegsrichtungen h_k, während der Gradient $grad\ f(x_k)$ die Richtung des steilsten Anstieges repräsentiert.
Die Güte eines Algorithmus hängt also wesentlich von drei Faktoren ab:

- Startpunkt x_0
- Richtung des steilsten Abstieges h_k
- Schrittweite λ_k

Die Algorithmen, die im Folgenden vorgestellt werden, unterscheiden sich in der Wahl dieser drei Parameter. Die Abstiegsrichtungen h_k können auch in der allgemeinen Form

$$h_k = -Q(x_k) \cdot grad\ f(x_k) \tag{5.37}$$

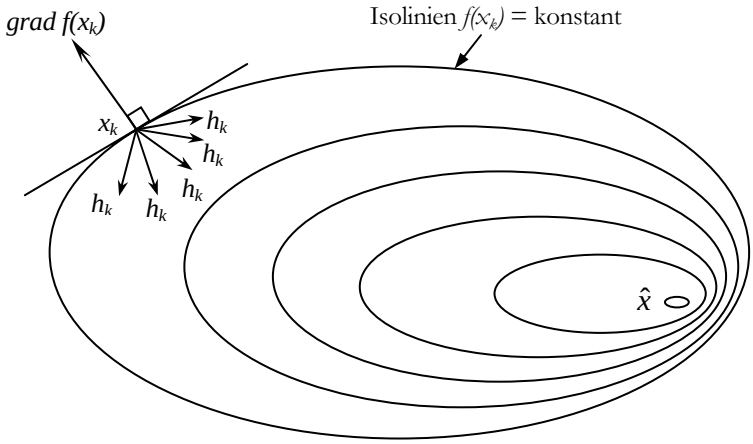


Abbildung 5.8: Abstiegsrichtungen im Punkt x_k.

geschrieben werden, wobei $Q(x_k)$ eine beliebige, positiv definite Matrix sei, die die Richtung des Abstieges (siehe Abb. 5.8) ändert. Durch Einsetzen von h_k in (5.36)erhält man die Iterationsformel für die gesamte Klasse der Methoden des steilen Abstieges in Form von

$$x_{k+1} = x_k - \lambda_k \cdot Q(x_k) \cdot grad\ f(x_k), \tag{5.38}$$

wobei sich die Verfahren lediglich in der Wahl der positiven Schrittweite λ_k und der positiv definiten Matrix $Q(x_k)$ unterscheiden. Die Konvergenz dieser Verfahren ist sowohl von dem funktionalen Zusammenhang als auch von Näherungswerten abhängig. Prinzipiell lassen sich beliebig viele Algorithmen entwickeln, die letztendlich eine Gemeinsamkeit haben: mittels Funktionsableitungen wird das funktionale Modell durch ein einfaches ersetzt. Einige dieser Verfahren, die eine gewisse Bedeutung im unseren speziellen Fall wegen ihrer Einfachheit und guter Konvergenz erlangt haben, werden im folgendem vorgestellt:

- Klassisches Abstiegsverfahren
- Newton Verfahren
- Levenberg – Marquardt Verfahren

Wird die Matrix Q in (5.38) als Einheitsmatrix gewählt, so spricht man von dem *klassischen Abstiegsverfahren* oder auch von dem Gradientenverfahren. Die Iterationsformel lautet dann

$$x_{k+1} = x_k - \lambda_k \cdot grad\ f(x_k). \tag{5.39}$$

Die klassischen Abstiegmethoden lassen sich je nach Wahl der Schrittweite λ_k weiter unterteilen. Dass die Wahl der Schrittweite nicht unproblematisch ist, zeigt [49]. Zu kleine Schrittweiten benötigen viele Iterationen oder konvergieren nicht. Das Gleiche gilt auch für zu große Schrittweiten. Am geläufigsten ist daher das Verfahren mit der Schrittweitenminimierungsregel: Auf der Halbgeraden in Richtung des stärksten Abstiegs wird nach dem Funktionsminimum gesucht. Ein Liniensuchalgorithmus liefert dann als eindimensionales Optimierungsproblem die optimale Schrittweite für den jeweiligen Nachfolger. Doch die bekannte Veranschaulichung eines zweidimensionalen Problems (Abb. 5.9), zeigt, dass durch die Orthogonalität aufeinander folgender Suchrichtungen ein Zickzack Kurs beschrieben wird. Je langgestreckter die Ellipse ist, desto mehr Iterationen werden

benötigt. Für höherdimensionale Probleme werden dann sehr viele Schritte erforderlich, wodurch das Verfahren an Effizienz verliert.

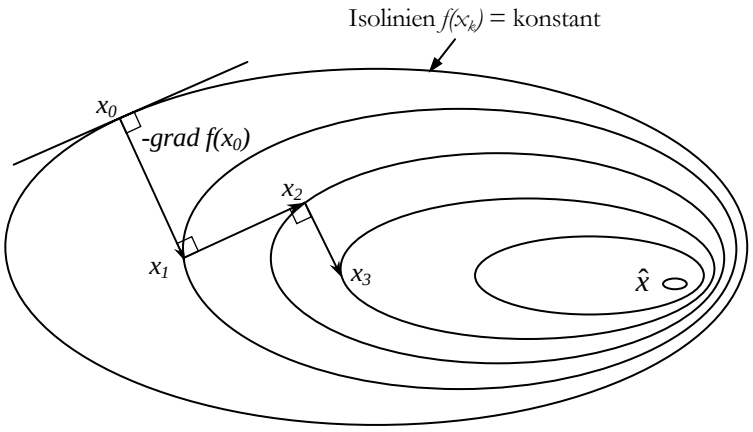


Abbildung 5.9: Abstieg des Gradientenverfahrens mit optimaler Schrittweite

Während die klassischen Gradientenverfahren die Fehlerfunktion linear ersetzen, wird beim Newton-Verfahren die Zielfunktion bei jedem Iterationsschritt quadratisch approximiert. Als Konsequenz konvergiert das Newton-Verfahren schnell, bei quadratischen Funktionen sogar in einem Schritt, man spricht hier von quadratischer Konvergenz.
Wie schon oben beschrieben wurde, laufen die Abstiegsmethoden zur Minimierung differenzierbarer Funktionen f für offene Mengen $V \in IR^m$ darauf hinaus, Punkte $x \in V$ zu bestimmen oder wenigstens anzunähern, für den

$$grad\ f(x) = O_m \tag{5.40}$$

ist, wobei O_m ein Nullvektor mit m - Elementen ist. Das bedeutet aber die Lösung eines (im Allgemeinen nichtlinearen) Gleichungssystems. Besitzt f in jedem Punkt $x \in V$ (stetige) partielle Ableitungen zweiter Ordnung $f_{x_i x_j}(x)$, $i,j=1,...,m$, so bietet sich zur Lösung von (5.40) das bekannte *Newtonische Abstiegsverfahren* an:

Ausgehend von irgendeinem Startpunkt $x_0 \in V$ wird eine Folge (x_k) in V derart zu konstruieren versucht, dass für jedes x_k mit nicht-singulärer Hesse-Matrix $H(x_k)$ der nächste Punkt x_{k+1} nach der Formel

$$x_{k+1} = x_k - H(x_k)^{-1} grad\ f(x_k) \tag{5.41}$$

berechnet wird.
Dabei ist jedoch im Allgemeinen nicht sichergestellt, dass $x_{k+1} \in V$ und $f(x_{k+1}) < f(x_k)$ ist. Die Forderung $x_{k+1} \in V$ lässt sich auf Grund der Offenheit von V durch Einführung eines geeigneten *Dämpfungsfaktors* λ_k erfüllen, d.h., anstelle von (5.41) rechnet man nach der Formel

$$x_{k+1} = x_k - \lambda_k H(x_k)^{-1} grad\ f(x_k). \tag{5.42}$$

Ist $|\lambda_k|$ genügend klein, so ist $x_{k+1} \in V$ gesichert. Die Bedingung $f(x_{k+1}) < f(x_k)$ ist erfüllbar, wenn die Hesse-Matrix $H(x_k)$ symmetrisch und positiv definit ist; denn dann ist $H(x_k)^{-1}$ ebenfalls symmetrisch und positiv definit, und für

$$h_k = -H(x_k)^{-1}\, grad\ f(x_k) \tag{5.43}$$

gilt

$$(h_k)^T\, grad\ f(x_k) = -grad\ f(x_k)^T\, H(x_k) grad\ f(x_k) < 0,$$

d.h. h_k ist eine Abstiegsrichtung. Wählt man dann $\lambda_k>0$ so, dass $x_k+\lambda_k h_k \in V$ ist und

$$f(x_k + \lambda_k h_k) \le f(x_k + \lambda h_k) \text{ für alle } \lambda>0 \text{ mit } x_k+\lambda h_k \in V \tag{5.44}$$

(sofern dies möglich ist), so folgt für

$$x_{k+1} = x_k + \lambda_k h_k,$$

dass $x_{k+1} \in V$ ist und $f(x_{k+1})<f(x_k)$.
Der Vorteil solcher Verfahren liegt in der schnellen quadratischen Konvergenzgeschwindigkeit. Dafür ist der numerische Aufwand relativ hoch gegenüber den Methoden erster Ordnung, denn in jeder Iteration muss die Hesse-Matrix $H(x_k)$ der Fehlerfunktion berechnet werden. Will man diesen Aufwand (der oft nicht unerheblich ist) vermeiden, so kann man auch das *vereinfachte Newtonische Abstiegsverfahren* durchführen.
Eine allgemeine Klasse von Abstiegsverfahren, die dem Newton-Verfahren ähnlich sind, lässt sich wie folgt gewinnen: Vorgegeben sei eine $m\times m$-Matrix-Funktion $Q=Q(x)$ derart, dass $Q(x)$ für jedes $x\in V$ symmetrisch und positiv definit ist. Dabei ist wiederum V eine nichtleere offene Teilmenge von IR^m, für die eine Funktion f (mit partiellen Ableitungen $f_{x_i}(x)$, $i=1,..., m$, für jedes $x\in V$) minimiert werden soll. Ist dann ein Startpunkt $x_0 \in V$ mit $grad\ f(x_0) \neq O_m$ vorgegeben, so ist

$$h = -Q(x_0)\, grad\ f(x_0) \tag{5.45}$$

eine Abstiegsrichtung; denn es ist

$$(h)^T\, grad\ f(x_0) = -grad\ f(x_0)^T\, Q(x_0) grad\ f(x_0) < 0.$$

Damit kann man das oben beschriebene Abstiegsverfahren mit Richtungen der Form (5.45) durchführen. Wir wollen das am allgemeinen Problem der nichtlinearen Ausgleichsrechnung demonstrieren. Hier ist

$$f(x) = y(x)^T \cdot y(x) = \sum_{j=1}^{n} y_j(x)^2, \tag{5.46}$$

und $y_1,...,y_n$ sind Funktionen auf einer offenen Teilmenge V von IR^m mit stetigen partiellen Ableitungen $y_{jx_i}(x), \begin{Bmatrix} i=1,...,m \\ j=1,...,n \end{Bmatrix}$ für alle $v\in V$. Damit erhält man

$$f_{x_i}(x) = \frac{df(x)}{dx_i} = 2 \cdot \sum_{j=1}^{n} y_j(x) \cdot y_{jx_i}(x), \qquad i = 1,...,m,$$

oder

$$grad\quad f(x) = 2 \cdot J(x)^T \cdot y(x) \tag{5.47}$$

wenn man die sog. *Jacobi-Matrix J(x)* wie folgt definiert:

$$J(x) = \begin{bmatrix} y_{1x_1}(x) & \cdots & y_{1x_m}(x) \\ y_{2x_1}(x) & \cdots & y_{2x_m}(x) \\ \vdots & & \vdots \\ y_{nx_1}(x) & \cdots & y_{nx_m}(x) \end{bmatrix}. \tag{5.48}$$

Besitzen $y_1,...,y_n$ in jedem Punkt $x \in V$ stetige zweite partielle Ableitungen $\frac{\partial^2 y_j}{\partial x_i \partial x_k}(x)$, $i,k=1,...,m$, $j=1,...,n$, so berechnet sich die Hesse-Matrix von f wie folgt: Für jedes Paar $i,k \in \{1,...,m\}$ ist

$$f_{x_i x_k}(x) = 2 \cdot \sum_{j=1}^{n} \left(y_{jx_i}(x) \cdot y_{jx_{k_i}}(x) + y_j(x) \cdot y_{jx_{i_i} x_k}(x) \right)$$

und somit

$$H(x) = 2J(x)^T J(x) + M(x) \tag{5.49}$$

wobei $M(x)$ eine $m \times m$-Matrix mit den Elementen

$$M_{ik}(x) = \sum_{j=1}^{n} \left(y_j(x) \cdot y_{jx_{i_i} x_k}(x) \right)$$

für $i,k=1,...,m$ ist. Die Matrix $2J(x)^T J(x)$ ist symmetrisch und positiv definit.
Dadurch wird folgendes Vorgehen im nichtlinearen Fall nahe gelegt: Anstelle der Hesse-Matrix $H(x)$ operiert man mit der Matrix $2J(x)^T J(x)$ (d.h. man vernachlässigt in (5.49) die Matrix $M(x)$). Da die Hesse- Matrix nicht notwendig positiv definit ist, ersetzt man sie durch eine Matrix der Form

$$L(x) = 2J(x)^T J(x) + \lambda I\,, \tag{5.50}$$

wobei I die $m \times m$ - Einheitsmatrix ist und λ eine positive reelle Zahl. Die durch (5.50) definierte Matrixfunktion ist dann für jedes $x \in V$ symmetrisch und positiv definit. Damit kann die oben genannte Matrixfunktion $Q=Q(x)$ zur Gewinnung eines Abstiegsverfahrens durch $Q(x)=L(x)^{-1}$ definiert werden. Um dieses Verfahren formulieren zu können, wird noch ein folgender Satz benötigt:

Zu einem vorgegebenen $x_0 \in V$ mit

$$grad\ f(x_0) = 2J(x_0)^T y(x_0) \neq O_m$$

sei $h=h(\lambda)$ die eindeutige Lösung des linearen Gleichungssystems

$$h = -\left(L(x_0)\right)^{-1} grad\ f(x_0),$$

oder $\left(2J(x_0)^T J(x_0) + \lambda I\right) h(\lambda) = -2J(x_0)^T y(x_0)$.

Dann ist für genügend großes $\lambda > 0$

$x_0 + h(\lambda) \in V$ und

$f(x_0 + h(\lambda)) < f(x_0)$.

Damit lässt sich das *Verfahren von Levenberg-Marquardt* formulieren: Wir wählen $x_0 \in V$ im $\lambda_0 > 0$, ein $\alpha > 0$ und setzen $k=0$. Damit gehen wir zu

1. Berechne

   $$grad\ f(x_k) = 2J(x_k)^T y(x_k).$$

   Ist $grad\ f(x_k) = O_m$, so bricht das Verfahren ab.

2. Andernfalls wird $h(\lambda_k) \in IR^m$ als Lösung von

   $$\left(2J(x_k)^T J(x_k) + \lambda_k I\right) h(\lambda_k) = -2J(x_k)^T y(x_k) \tag{5.51}$$

   berechnet. Ist

   $$x_k + h(\lambda_k) \in V \text{ und} \tag{5.52}$$

   $$f(x_k + h(\lambda_k)) < f(x_k),$$

   so wird k durch $k+1$ und $\lambda_{k+1} = \lambda_0$ sowie $x_{k+1} = x_k + h(\lambda_k)$ gesetzt und nach 1. gegangen.
   Ist $x_k + h(\lambda_k) \notin V$ und $f(x_k + h(\lambda_k)) \geq f(x_k)$,
   so wird λ_k durch $\alpha\lambda_k$ $(\alpha > 1)$ ersetzt und nach 2. gegangen.

Auf Grund von vorgegebenem Satz wird nach endlich vielen Erhöhungen $\alpha\lambda_k$ die Bedingung (5.51) erfüllt sein.
Aus oben beschriebenen Algorithmus der Berechnung, ergibt sich nach [50] die Iterationsformel (5.53) für das Levenberg-Marquardt-Verfahren:

$$x_{k+1} = x_k - \left(2J(x_k)^T \cdot J(x_k) + \lambda_k \cdot I\right)^{-1} \cdot grad\ f(x_k) \tag{5.53}$$

Die Formel selbst zeigt schon die beiden Grundprinzipen, auf denen das Verfahren basiert: Da I eine positiv definiten Einheitsmatrix ist, kann immer ein hinreichend großes λ_k gefunden werden, so dass $\left(2J(x_k)^T \cdot J(x_k) + \lambda_k \cdot I\right)^{-1}$ in (5.53) positiv definit ist. Damit ist immer eine Abstiegsrichtung garantiert. Dadurch kann ein Mangel des Newton-Verfahrens, nämlich der einer eventuell nicht positiv definiten Hesse-Matrix $H(x)$, behoben werden. Andererseits zeigt (5.50), dass das Verfahren ein Kompromiss zwischen dem klassischen Abstiegsverfahren und dem Newton-Verfahren darstellt. Wählt man in (5.53) $\lambda_k = 0$, so erhält man die Formel (5.41) des Newton-Verfahrens. Für sehr große λ_k geht das Verfahren in die klassische Abstiegsmethode (5.39) über. Die möglichen Suchrichtungen liegen also zwischen der des Newton- und der des klassischen Abstiegsverfahrens (Abb. 5.10).
Einerseits ist LM-Algorithmus wesentlich flexibler als die anderen oben genannten Verfahren, andererseits bleibt aber die Frage offen, welchen Wert λ_k annehmen soll. Da es nicht möglich ist, für alle Optimierungsprobleme eine allgemeine Regel zu formulieren, bleibt nichts anderes übrig, als einen selbst gewählten Wert für λ_k dahingehend zu testen, ob eine Abstiegsrichtung vorliegt oder nicht. Ist $\left(2J(x_k)^T \cdot J(x_k) + \lambda_k \cdot I\right)^{-1}$ in (5.53) positiv definit, liegt also eine Abstiegsrichtung vor, so kann λ_k akzeptiert werden, anderenfalls muss ein erneuter Test mit einem größeren λ_k durchgeführt werden.
Das Levenberg-Marquardt-Verfahren (LM-Verfahren) vereint also die Vorteile beider Methoden – die Einfachheit der klassischen Abstiegsmethoden und die guten

Konvergenzeigenschaften des Newton-Verfahrens. Dabei wurden die Nachteile beider Methoden eliminiert – die schlechte Konvergenz des klassischen Abstiegsverfahrens und die Notwendigkeit der Berechnung der Hesse-Matrix *H(x)* im Newton-Verfahren.
Für unseren Fall ist das LM-Verfahren am besten geeignet, da im Gegensatz zum Newton-Verfahren die Schrittweiten variiert werden können, was sicher zum nächsten lokalen Minimum führt. Beim Einsatz des Newton-Verfahrens kann man das nächste Minimum verpassen, da die Schrittweite dieser Methode bei langsamem Verlauf der Robotertrajektorie bereits zu groß sein kann. Wenn die Trajektoriengeschwindigkeit dagegen hoch ist, kann unter Nutzung des klassischen Abstiegsverfahrens das lokale Minimum für den nächsten Punkt dieser Trajektorie wegen schlechter Konvergenz solcher Methode nie erreicht werden, da der Abstand zwischen aufeinander folgenden Punkten zu groß wird.

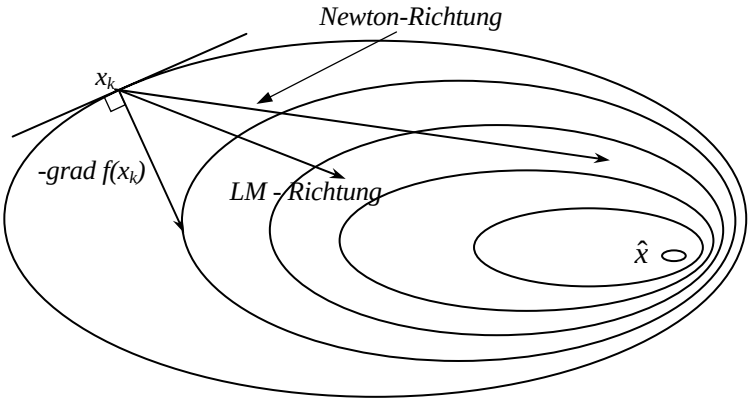


Abbildung 5.10: Suchrichtungen des Levenberg-Marquardt-Verfahrens

Je nach Trajektorienverlauf liefert sowohl das klassische Abstiegsverfahren als auch das Newton- Verfahren schlechte Lösung oder versagt gänzlich.
Wegen der sehr schnellen Konvergenz und relativen Einfachheit wurde für die Lösung unserer nichtlinearen Fehlerfunktion zur Bestimmung der inversen Kinematik das Levenberg-Marquardt-Verfahren ausgewählt. Es wird in den folgenden Abschnitten näher untersucht.

5.2.3 Verfahren zur Ermittlung der inversen Kinematik

Wie schon im Abschnitt 5.2.1 beschrieben wurde, werden in dieser Arbeit zwei Methoden für die Bestimmung der inversen Kinematik betrachtet, die sowohl Allgemeingültigkeit als auch Effizienz aufweisen. Die ***1. Methode*** bestimmt die Winkel der einzelnen Gelenke mit Hilfe des oben beschriebenen (Levenberg-Marquardt) Optimierungsverfahrens [50] für das nichtlineare Gleichungssystem(5.28), das für diese Optimierungsmethode in quadratischer Form:

$$E(\theta_n) = \sum_{i=0}^{6} \left((p_x, p_y, p_z, \alpha, \beta, \gamma) - f_i(\theta_n) \right)^2 \tag{5.54}$$

definiert werden muss. Wie schon im Abschnitt 5.2.2 beschrieben, sorgt diese quadratische Form der Fehlerfunktion für eine schnelle Konvergenz mittels dieser Optimierung. Dabei dienen die zuletzt ermittelten Gelenkwinkel als Startwerte für die Berechnung der sich anschließenden Winkel. Ausgehend von Anfangsposition und -Orientierung des Endeffektors

wird die nächste Position und Orientierung gemäß einer bestimmten Trajektorie mittels des Levenberg-Marquardt Optimierungsverfahrens ermittelt. Da das Levenberg-Marquardt Verfahren im allgemeinen Fall zufällige Anfangswerte verwendet, führt es zu einem großen Zeitaufwand. In unserem Fall sind die am Anfang bestimmten Achswinkel sowie die Position und Orientierung des Endeffektors bekannt. Da die Zykluszeit der offenen Robotersteuerung klein ist, werden die Positionen des Endeffektors in den kartesischen Koordinaten im Zeitraum zwischen zwei aufeinander folgenden Tastpunkten nicht wesentlich verändert, was zur geringen Veränderungen der Achswinkel führt. Deshalb sind nur wenige Schritte zur Optimierung der Gelenkwerte für jede Lage des Endeffektors erforderlich. Dies wird im folgenden Abschnitt 5.2.4 mit Hilfe der experimentellen Untersuchungen bekräftigt.

Gemäß (5.4) und (5.13) ergibt sich die Position und Orientierung des Endeffektors ${}^0T_7$ für den 7-achsigen Roboter *amtec r7*:

$$ {}^0T_7 = f(\theta_n) \tag{5.55} $$

oder unter Einbeziehung der Eulerschen Winkel:

$$ (p_x, p_y, p_z, \alpha, \beta, \gamma)^T = f(\theta_n), \tag{5.56} $$

wo p_x, p_y, p_z und (α, β, γ) entsprechend die Position und Orientierung des Endeffektors sind.

Für die Menge von Testsignalen zu den Abtastzeitpunkten *i* lässt sich die Fehlerfunktion in unserem Fall bestimmen:

$$ E(\theta_{n\,i}) = \sum \left({}^0T_{n\,i} - f_{m\,i-1}(\theta_{n\,i-1}) \right)^2 \tag{5.57} $$

Diese Fehlerfunktion ist eine nichtlineare Funktion und soll mit Hilfe des Levenberg-Marquardt Verfahrens minimiert werden. Die dazu notwendige Vorgehensweise zur Minimierung der Fehlerfunktion nach dem LM-Verfahren wurde bereits im Abschnitt 5.2.2 beschrieben. Gemäß (5.48) erhält man:

$$ J(\theta) = \frac{\partial f}{\partial \theta} = \begin{bmatrix} \frac{\partial f_1}{\partial \theta_1} & \cdots & \frac{\partial f_1}{\partial \theta_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial \theta_1} & \cdots & \frac{\partial f_m}{\partial \theta_n} \end{bmatrix} \tag{5.58} $$

und

$$ grad\ E(\theta_n) = 2J^T(\theta) \cdot y(\theta) = 2 \cdot \begin{bmatrix} \frac{\partial f_1}{\partial \theta_1} & \cdots & \frac{\partial f_1}{\partial \theta_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial \theta_1} & \cdots & \frac{\partial f_m}{\partial \theta_n} \end{bmatrix}^T \cdot \left({}^0T_n - f_m(\theta_n) \right) \tag{5.59} $$

Als Anfangswerte der Gelenkwinkel werden $\theta_{n0}=(\theta_{10}, \theta_{20},..., \theta_{n0})$ ausgewählt, das entspricht dem Anfangspunkt P_0 des Roboters in kartesischen Koordinatenraum. Die Berechnungen der neuen Werte der Gelenkwinkel werden gemäß der Gleichung (5.53) durchgeführt:

$$ x_{k+1} = x_k - \left(2J(x_k)^T \cdot J(x_k) + \lambda_k \cdot I \right)^{-1} \cdot grad\ f(x_k) \tag{5.60} $$

Aber es ist nicht leicht, für die Funktion (5.57) eine Jacobi-Matrix (5.58) zu finden. Dabei ist besonders die Bestimmung der Jacobi-Matrix für die Orientierung kompliziert, weil die Gleichungen (5.21, 5.24-5.25) eine *arctan-* Funktion enthalten, die eine komplizierte Ableitungsfunktion besitzt. Außerdem ist es wünschenswert bzw. erforderlich, diese Problematik numerisch zu lösen. Die Jacobi-Matrix wird durch Differenzierung der Gleichung (5.48) berechnet.

$$\frac{\partial y}{\partial t} = \frac{\partial (F(q))}{\partial t} = \frac{\partial (F(q))}{\partial t} \cdot \frac{\partial q}{\partial q} = \frac{\partial (F(q))}{\partial q} \cdot \frac{\partial q}{\partial t} = J(q) \cdot \frac{\partial q}{\partial t} \tag{5.61}$$

J(q) ist hier eine *6×n* Jacobi-Matrix, *y* ist ein *6×1* Spaltenvektor, bestehend aus Positionsvektor $y_p=p=[p_x, p_y, p_z]$ und Orientierungsvektor $y_0=\Phi=[\alpha, \beta, \gamma]$. Die Gleichung (5.61) kann man auch in folgender Form nach [28] aufschreiben

$$\begin{bmatrix} v(t) \\ \Omega(t) \end{bmatrix} = J_0(q) \cdot \dot{q}(t)\,, \tag{5.62}$$

wo *v(t)* die lineare Positionsgeschwindigkeit

$$v(t) = \frac{dp(t)}{dt} = \left[\frac{dp_x(t)}{dt}, \frac{dp_y(t)}{dt}, \frac{dp_z(t)}{dt} \right]^T = \left[v_x(t), v_y(t), v_z(t) \right]^T \tag{5.63}$$

und Ω(t) die Winkelgeschwindigkeit

$$\Omega(t) = \left[w_x, w_y, w_z \right]^T \tag{5.64}$$

ist. Dabei lässt sich die Relation zwischen $[w_x(t), w_y(t), w_z(t)]$ und $[\dot{\alpha}(t), \dot{\beta}(t), \dot{\gamma}(t)]$ nach Fu [28] durch folgende Gleichung finden:

$$\begin{bmatrix} w_x(t) \\ w_y(t) \\ w_z(t) \end{bmatrix} = \begin{bmatrix} \cos\gamma\cos\beta & -\sin\gamma & 0 \\ \sin\gamma\cos\beta & \cos\gamma & 0 \\ -\sin\beta & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \dot{\alpha}(t) \\ \dot{\beta}(t) \\ \dot{\gamma}(t) \end{bmatrix} = S(\Phi) \cdot \begin{bmatrix} \dot{\alpha}(t) \\ \dot{\beta}(t) \\ \dot{\gamma}(t) \end{bmatrix} \tag{5.65}$$

Die inverse Relation der Gl. (5.65) kann man durch

$$\begin{bmatrix} \dot{\alpha}(t) \\ \dot{\beta}(t) \\ \dot{\gamma}(t) \end{bmatrix} = \frac{1}{\cos\beta} \cdot \begin{bmatrix} \cos\gamma & \sin\gamma & 0 \\ -\sin\gamma\cos\beta & \cos\gamma\cos\beta & 0 \\ \cos\gamma\sin\beta & \sin\gamma\sin\beta & \cos\beta \end{bmatrix} \cdot \begin{bmatrix} w_x(t) \\ w_y(t) \\ w_z(t) \end{bmatrix} = S^{-1}(\Phi) \cdot \begin{bmatrix} w_x(t) \\ w_y(t) \\ w_z(t) \end{bmatrix} \tag{5.66}$$

finden, oder in der Matrix- Vektor Form:

$$\dot{\Phi}(t) = S^{-1}(\Phi) \cdot \Omega(t) \tag{5.67}$$

Mit dieser Korrektur für die Orientierung erhält die Gleichung (5.62) folgendes aussehen:

$$\begin{bmatrix} v(t) \\ \dot{\Phi}(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & S^{-1}(\Phi) \end{bmatrix} \cdot J_0(q) \cdot \dot{q}(t) = J(q) \cdot \dot{q}(t) \tag{5.68}$$

Durch Gleichung (5.68) lässt sich die Jacobi-Matrix *J(q)* numerisch bestimmen.
Zur Bestimmung der Jacobi-Matrix $J_0(q)$ in (5.62) und (5.68) existieren verschiedene Möglichkeiten, die in [28] und [70] angeführt sind. Eine davon, die für unseren Fall besonders

gut geeignet ist, da alle anderen Methoden mehr Rechenaufwand erfordern, wird hier näher betrachtet. Diese Methode verwendet das Kreuzprodukt von Vektoren. Dabei wird die *i*- te Spalte der Jacobi-Matrix $J_0(q)$ nach Fu [28] wie folgt berechnet:

$$J_{0i}(q) = \begin{cases} \begin{bmatrix} z_{i-1} \times \left({}^0p_n - {}^0p_{i-1} \right) \\ z_{i-1} \end{bmatrix} & \text{bei Drehgelenken} \\ \begin{bmatrix} z_{i-1} \\ 0 \end{bmatrix} & \text{bei Schubgelenken} \end{cases} \tag{5.69}$$

wo z_{i-1} ein Orientierungsvektor des *(i-1)*-ten Koordinatensystem bezogen auf das Basiskoordinatensystem ist, das aus der Gl. (5.4) zu ermitteln ist. ${}^0p_n$ ist ein Positionsvektor des Endeffektors zum Basiskoordinatensystem (0) und ${}^0p_{i-1}$ ist ein Positionsvektor der *(i-1)*-ten Koordinatensystemursprung zum Basiskoordinatensystem, die sich aus der Gl. (5.5) ermitteln lassen. Das Kreuzprodukt von zwei Vektoren in der Gl. (5.69) ergibt sich aus:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \times \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} x_2y_3 - x_3y_2 \\ x_3y_1 - x_1y_3 \\ x_1y_2 - x_2y_1 \end{bmatrix} \tag{5.70}$$

Ein Vorteil dieses Verfahrens im Gegensatz zu den anderen Verfahren zur Bestimmung der inversen Kinematik besteht darin, dass es die Ermittlung der Konfiguration (Gelenkwinkel) aus der vorgegebenen Position und Orientierung des Endeffektors für redundante Roboter numerisch effizient und im *Online*-Betrieb ermöglicht. Dabei wird ein genügend schneller Prozessor zur *Online*-Optimierung für jeden Punkt der vorgegebenen Trajektorie erforderlich, was ein gewisser nicht zu vermeidender Nachteil der vorliegenden Methode ist.
Aber nicht alle mathematisch errechneten Lösungen aufgrund der LM-Optimierung sind auch nutzbare Lösungen, da die Gelenkwinkelbegrenzungen des Roboters berücksichtigt werden müssen und weitere Lösungen darüber hinaus zu Kollisionen führen würden. Deswegen wurde noch eine Überwachungsfunktion neben den 6 Gleichungen für Position und Orientierung des Endeffektors eingeführt.

$$E(\theta_{n\,i}) = \sum \left({}^0T_{ni} - f_{n\,i-1}(\theta_{n\,i-1}) \right)^2 + \sum E_u(\theta_{n\,i-1}) \tag{5.71}$$

Diese Funktion $\sum E_u(\theta_n)$ bildet die Summe einzelner Funktionen, die die Grenzwerte (siehe Tabelle 5.3) für jedes Gelenk des Roboters berücksichtigen. Jede einzelne Funktion ist dabei eine Fehlerfunktion, deren Werte des Fehlers bei kleinen Gelenkwinkeln nahezu Null sind und bei Erreichen der Grenzwerte stark ansteigen (siehe Abb. 5.11).

Max. Winkel	$\theta_1=\pm 160°$	$\theta_2=\pm 95°$	$\theta_3=\pm 160°$	$\theta_4=\pm 109°$	$\theta_5=\pm 160°$	$\theta_6=\pm 95°$	$\theta_7=\pm 360°$
$E_u(\theta_n)$	$0.1\cdot\left(\frac{\theta_1}{160}\right)^{20}$	$0.1\cdot\left(\frac{\theta_2}{95}\right)^{20}$	$0.1\cdot\left(\frac{\theta_3}{160}\right)^{20}$	$0.1\cdot\left(\frac{\theta_4}{109}\right)^{20}$	$0.1\cdot\left(\frac{\theta_5}{160}\right)^{20}$	$0.1\cdot\left(\frac{\theta_6}{95}\right)^{20}$	$0.1\cdot\left(\frac{\theta_7}{360}\right)^{20}$

Tabelle 5.3: Die Überwachungsfunktion für jeden Gelenk des Roboters

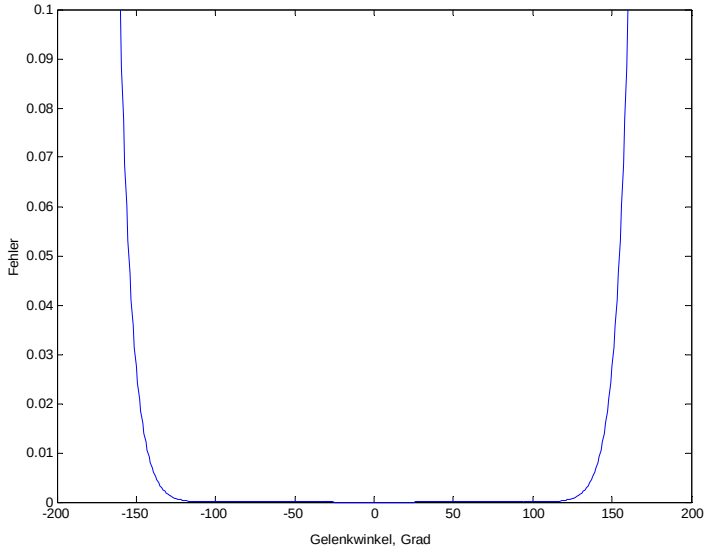


Abbildung 5.11: Die Kostenfunktion für den Gelenkwinkel θ_1

Bei der Annäherung der Winkelwerte eines Gelenkes an einen seiner Grenzwerte wird das Levenberg-Marquardt- Optimierungsverfahren den Verlauf des Gelenkwinkels aufgrund der Berücksichtigung der Überwachungsfunktion in das sichere Gebiet umkehren, weil der gesamte Fehler in Gl. (5.71) sonst zu groß wird. Das LM-Verfahren muss also eine Fehlerfunktion (5.71) mit den Erweiterungen in Form der so genannten Kostenfunktionen (Abb. 5.11) für jedes einzelnes Gelenk minimieren. Ist darüber hinaus das (Selbst-) Kollisionsproblem zu betrachten, können anderen Überwachungsfunktionen notwendig werden.
Der Rechenaufwand für unseren Anwendungsfall zur *Online*-Berechnung der inversen Kinematik mit dem im Kapitel 3 ausgewählten DSpace-Mehrprozessorsystem ist vertretbar, dies wird im Abschnitt 5.2.4 experimentell mit einer Zykluszeit von 10ms nachgewiesen.

Die ***2. Methode*** repräsentiert mit Hilfe eines neuronalen Netzes (RBF bzw. SOM), das in [37] beschrieben wurde, die Parameter einer inversen Jacobimatrix $J^{-1}(q)$. Dazu kann eine Vektorfunktion y von n Variablen aufgestellt werden

$$\begin{aligned} y_1 &= f_1(x_1, x_2, \ldots, x_n) \\ y_2 &= f_2(x_1, x_2, \ldots, x_n) \\ &\vdots \\ y_2 &= f_2(x_1, x_2, \ldots, x_n) \end{aligned} \tag{5.72}$$

Wobei n für die Anzahl der Achsen steht. Diese Gleichungen können als

$$y = F(x) \tag{5.73}$$

beschrieben werden. Das Differential dy/dt kann wie folgt berechnet werden:

$$\frac{\partial y}{\partial t} = \frac{\partial (F(x))}{\partial t} = \frac{\partial (F(x))}{\partial t} \cdot \frac{\partial x}{\partial x} = \frac{\partial (F(x))}{\partial x} \cdot \frac{\partial x}{\partial t} \tag{5.74}$$

Somit gilt:

$$\begin{aligned}\dot{y}_1 &= \frac{\partial f_1}{\partial x_1}\dot{x}_1 + \frac{\partial f_1}{\partial x_2}\dot{x}_2 + \cdots + \frac{\partial f_1}{\partial x_n}\dot{x}_n \\ &\vdots \\ \dot{y}_n &= \frac{\partial f_n}{\partial x_1}\dot{x}_1 + \frac{\partial f_n}{\partial x_2}\dot{x}_2 + \cdots + \frac{\partial f_n}{\partial x_n}\dot{x}_n\end{aligned} \tag{5.75}$$

Damit lässt sich $\frac{\partial F}{\partial x}$ in Matrizenform darstellen:

$$J(x) = \frac{\partial F}{\partial x} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1} & \cdots & \frac{\partial f_n}{\partial x_n} \end{bmatrix} \tag{5.76}$$

Die entstehende *(n,n)* Matrix ist die gesuchte Jacobimatrix *J*. Sind die Funktionen f_i in x nichtlinear, wie in unserer Falle, so ist *J* eine Funktion von x. Es gilt also:

$$\dot{y} = J(x) \cdot \dot{x} \tag{5.77}$$

Setzt man für x die Gelenkvariablen

$$(\theta_1, \ldots, \theta_n) = q = x \tag{5.78}$$

und für *y* die Position und Orientierung des Endeffektors

$$(p_x, p_y, p_z, \alpha, \beta, \gamma) = s = y \tag{5.79}$$

ein, kann die Jacobimatrix zur Transformation von kartesischen Koordinaten in Achskoordinaten benutzt werden. Es kann folgende Gleichung aufgestellt werden:

$$\frac{\partial s}{\partial t} = J(q) \cdot \frac{\partial q}{\partial t} \tag{5.80}$$

Sind die Achskoordinaten *q* gesucht, so lassen sich diese über die Bewegungsdifferenzen aus der Beziehung

$$\frac{\partial q}{\partial t} = J^{-1}(q) \cdot \frac{\partial s}{\partial t} \tag{5.81}$$

oder $$\partial q = J^{-1}(q) \cdot \partial s$$

berechnen. Die Geschwindigkeiten können mit

$$\begin{aligned}\frac{\partial s}{\partial t} &= (\dot{p}_x, \dot{p}_y, \dot{p}_z, \omega_x, \omega_y, \omega_z) \\ \frac{\partial q}{\partial t} &= (\dot{\theta}_1, \dot{\theta}_2, \ldots, \dot{\theta}_n)\end{aligned} \tag{5.82}$$

berechnet werden. Wobei die Faktoren $(\dot{p}_x, \dot{p}_y, \dot{p}_z)$ und $(\omega_x, \omega_y, \omega_z)$ die Positionsgeschwindigkeiten und die lineare Winkelgeschwindigkeit in kartesischen Koordinaten beschreiben.

In diesem Fall kann man mit Hilfe eines neuronalen Netzes (RBF oder SOM) die inverse Jacobimatrix $J^{-1}(q)$ (siehe Gl. 5.81) repräsentieren bzw. lernen. Dieser Lösungsweg ist allgemein, jedoch weniger effizient im Vergleich zur geschilderten 1. Methode. Dabei ist auf folgende Nachteile dieser Methode hinzuweisen:

1. Sehr großer Speicher- und Zeitaufwand, da die Notwendigkeit zur Nutzung großer Menge von Daten (die Position und die Orientierung des Endeffektors im Arbeitsraum) für das erfolgreiche Training des Netzes entsteht. Dabei sind solche Trajektorien zu generieren, die durch alle Punkte des Arbeitsraumes des Roboters mit verschiedener Orientierung des Endeffektors laufen. Die Ausführung solcher Trajektorien führt als Folge zur Abnutzung des Roboters. Da das entstehende Netz aus Genauigkeitsgründen mindestens 10 Neuronen für jeden der 6 Freiheitsgrade umfassen muss, wird die Gesamtzahl der Neuronen in diesem Fall 10^6=1000000 betragen. Für jedes dieser Neuronen muss die Jacobi- Matrix (5.76) trainiert werden, die eine Größe *6x7* hat. Daraus ist ersichtlich, dass diese Methode der Berechnung der inversen Kinematik auch für heutige moderne Rechner noch sehr rechen- und speicheraufwändig ist.
2. Nach dem Training des Netzes für jeden Wert der Position und Orientierung des Endeffektors existiert mitunter nicht nur eine Konfiguration des Roboters. Dies führt zur Notwendigkeit der Auswahl einer passenden Konfiguration (siehe Abb. 5.7).

Nachdem das Netz trainiert wurde, existiert keine Notwendigkeit der Nutzung der schnellen Prozessoren mehr, da für bestimmte Position und Orientierung des Endeffektors das neuronale Netz die entsprechenden Gelenkwinkel des Roboters sofort findet. Es sei denn, das Training bzw. Lernen des Netzes wird während der Arbeit des Gesamtverfahrens fortgesetzt.

Da die erste Methode fast alle Nachteile der zweiten Methode eliminiert, wurde das LM-Optimierungsverfahren für die Lösung der Aufgabe der inversen Kinematik gewählt. Dabei kann man den großen Rechenaufwand für die *Online*-Berechnung der ersten Methode vernachlässigen, da die moderne Hardware genügend schnelle Prozessoren bereitstellen. Die Aufgabe der Ermittlung der inversen Kinematik mit Hilfe der Levenberg-Marquardt - Optimierung ist für den 7-achsigen Roboter besonders interessant, da die üblichen Lösungsmethoden keine eindeutige Lösungen finden können, weil das Gleichungssystem (5.54) für den 7-achsigen Roboter unterbestimmt ist, also die Anzahl der Gleichungen (6) (Position und Orientierung des Endeffektors) ist kleiner als die Anzahl der Unbekannten (7 Gelenkwinkel).

5.2.4 Simulative und experimentelle Untersuchungen

Nachdem im Abschnitt 5.2.3 betrachteten Vergleich aller Vor- und Nachteile beider Methoden zur Ermittlung der inversen Kinematik wurde entschieden, dies mittels des Levenberg-Marquardt-Optimierungsverfahrens durchzuführen. Die Anwendbarkeit und Effektivität dieser entwickelten Methode wird im folgenden sowohl simulativ als auch experimentell für den Roboter *amtec r7* mit 7 Achsen nachgewiesen. Gemäß der bereits oben beschriebenen Vorgehensweise zur Berechnung der inversen Kinematik wurde das C-Programm „LMoptim7.c“ geschrieben, das aus der Position (P_x, P_y, P_z) und der Orientierung (α, β, γ) die 7 Gelenkwinkel des Roboters berechnet. Um zu ermitteln, ob das Programm fehlerfrei arbeitet, wurden zuerst simulative Untersuchungen durchgeführt. Dazu wurde mit

Hilfe der Simulink-Bibliothek des Matlabs ein Signalflussbild (siehe Abb. 5.12) entwickelt, das aus Trajektoriengenerator, inverser Kinematik und sich anschließender direkter Kinematik besteht.

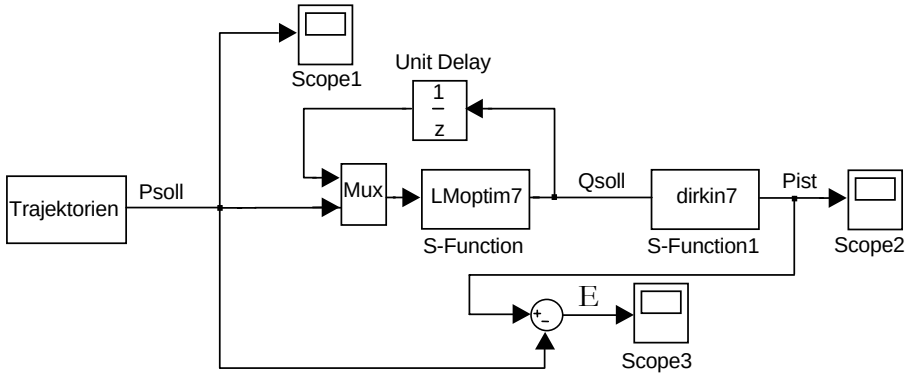


Abbildung 5.12: Signalflussbild zur simulativen Untersuchungen für den Roboter „amtec r7“

Der Trajektoriengenerator liefert räumliche Trajektorien (P_{soll}), die dann mittels der inversen Kinematik in die entsprechenden Gelenkwinkel des Roboters (Q_{soll}) transformiert werden. Anschließend transformiert die direkte Kinematik diese Winkel in der Position und Orientierung des Endeffektors (P_{ist}). Aus der Differenz zwischen P_{soll} und P_{ist} wird der quadratische Fehler (E) gebildet, der bei der Berechnung der inversen Kinematik entsteht.

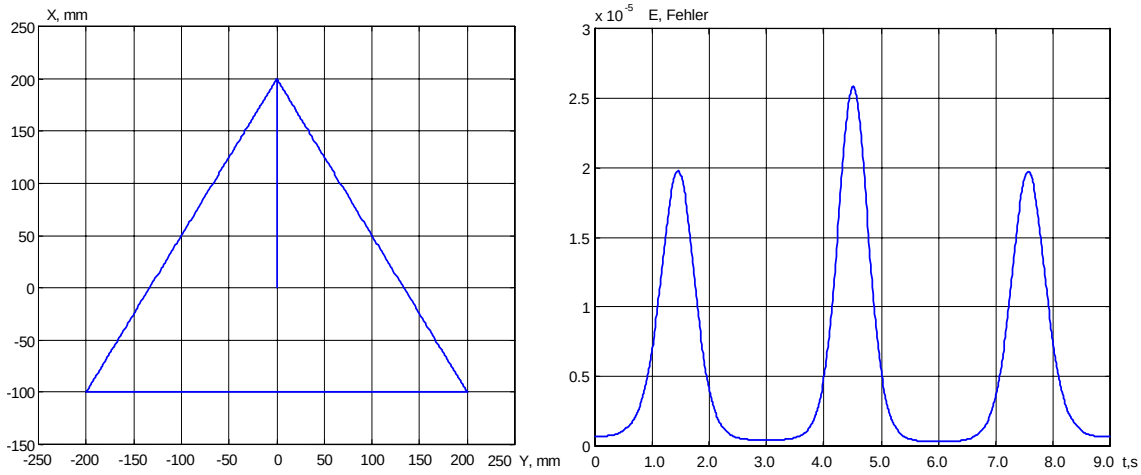


Abbildung 5.13: Planare dreieckförmige Trajektorie im Raum (links) und entsprechend die quadratischen Fehler bzgl. der Lage des Endeffektors (rechts)

Zur Testung der gewählten Variante der Realisierung der inversen Kinematik wurde zunächst eine planare dreieckförmige Trajektorie im Raum mit konstantem Sollwert für die Z-Koordinate gestaltet, die in Abb. 5.13 links dargestellt ist. Rechts ist entsprechend die Summe aller quadratischen Fehler nach Gl. (5.71) bezüglich der Position und der Orientierung des Endeffektors in der XY-Ebene aufgetragen. Es wurden die linearen Trajektorien zwischen den Punkten im Arbeitsraum gestaltet. Die gewünschte Raumkurve wird erzeugt, in dem der Endeffektor von einem Punkt im Raum zu dem anderen gebracht wird. Die genaue Beschreibung und die Form solcher Trajektorie wurden im Kapitel 4 dargestellt. Wie man sieht, wächst der Fehler bei der Erhöhung der Geschwindigkeit (in der Mitte) der vorgegebenen Trajektorie an. Für jeden Punkt dieser dreieckförmigen Trajektorie wurden

mittels LM-Optimierung in jedem Steuertakt die entsprechenden Gelenkwinkel gefunden. Nach 10 Optimierungsschritten liegen sie mit ausreichender Genauigkeit (siehe Abb. 5.13) vor. Weiterhin ist es erforderlich, den Trajektorienverlauf in der Nähe singulärer Punkte zu untersuchen, um zu zeigen, dass die vorgeschlagene Methode der Berechnung der inversen Kinematik in diesen Punkten keine Ergebnisse liefert, die zur heftigen Umorientierungen des Roboters führen. Dafür wurde das Signalflussbild nach Abb. 5.14 entwickelt und simulativ getestet. Dabei dienen als Trajektorien sinusförmigen Kurven für jedes Gelenk (Q_{soll}) des Roboters (siehe Abb. 5.15). Gemäß diesen Trajektorien in den Gelenkwinkeln berechnet die direkte Kinematik entsprechende Sollposition und -orientierung des Endeffektors (P_{soll}). Dann transformiert sie die inverse Kinematik in die Istgelenkwinkel (Q_{ist}).

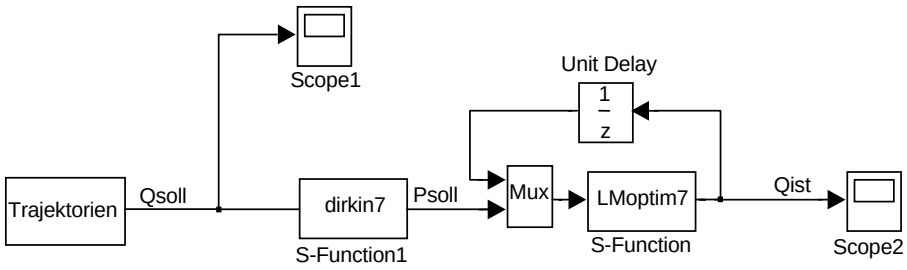


Abbildung 5.14: Signalflussbild zur simulativen Untersuchungen für den Roboter „amtec r7"

Aus der Abb. 5.15 ist ersichtlich, dass die inverse Kinematik fast denselben Trajektorienverlauf der Gelenkwinkel (rechts), wie die vorgegebene sinusförmige Eingangssignale für die direkte Kinematik (links) liefert. Den einzigen Unterschied zwischen den beiden Signalverläufen stellt man in der Nähe von einem singulären Punkt fest, wo alle Gelenkwinkel nahezu Null werden. Wie man aus der Abb. 5.15 sieht, wurde der singuläre Punkt umgegangen, so dass die Gelenkwinkel Werte verschieden von Null annehmen. Dabei zeigt die Abbildung 5.16 den Verlauf der Fehlerfunktion für die Position und Orientierung des Endeffektors in solch einem singulären Punkt. Wie man sieht, findet die inverse Kinematik für den 7-achsigen Roboter geringfügig veränderte Lösungen als die vorgegebenen Trajektorien der Gelenkwinkel, was sich als Auswirkung der 7. Achse erklären lässt. Die Gelenkwinkel für die Achsen 1, 3, 5 und 7 wurden gleichzeitig minimal verändert. D.h., dass das LM-Optimierungsverfahren für die Berechnung der inversen Kinematik die Gelenkwinkel des Roboters für alle Achsen minimal entlang einer gegebenen Trajektorie im Arbeitsraum (Position und Orientierung) verändert.

Es ist ebenfalls wichtig zu untersuchen, welche Konfiguration der Roboter in der Nähe von Grenzwerten der Gelenke einnimmt. Dazu wurde wiederum das Signalflussbild der Abb. 5.12 herangezogen. Als Eingangstrajektorie wurde die in Abb. 5.13 dargestellte dreieckförmige Trajektorie im kartesischen Arbeitsraum verwendet. Es wurden die Verläufe der Gelenkwinkel am Ausgang der inversen Kinematik mit und ohne Berücksichtigung der Überwachungsfunktionen für jedes einzelnes Gelenk des Roboters untersucht (siehe Abb. 5.17). Daraus ist ersichtlich, dass bei Annäherung der Winkelwerte des 1. Gelenkes an einen seiner Grenzwerten (-160°÷+160°) das Levenberg-Marquardt-Optimierungsverfahren den Verlauf des Gelenkwinkels aufgrund der Berücksichtigung der Überwachungsfunktion in das zulässige und praktisch realisierbare Gebiet kehrt.

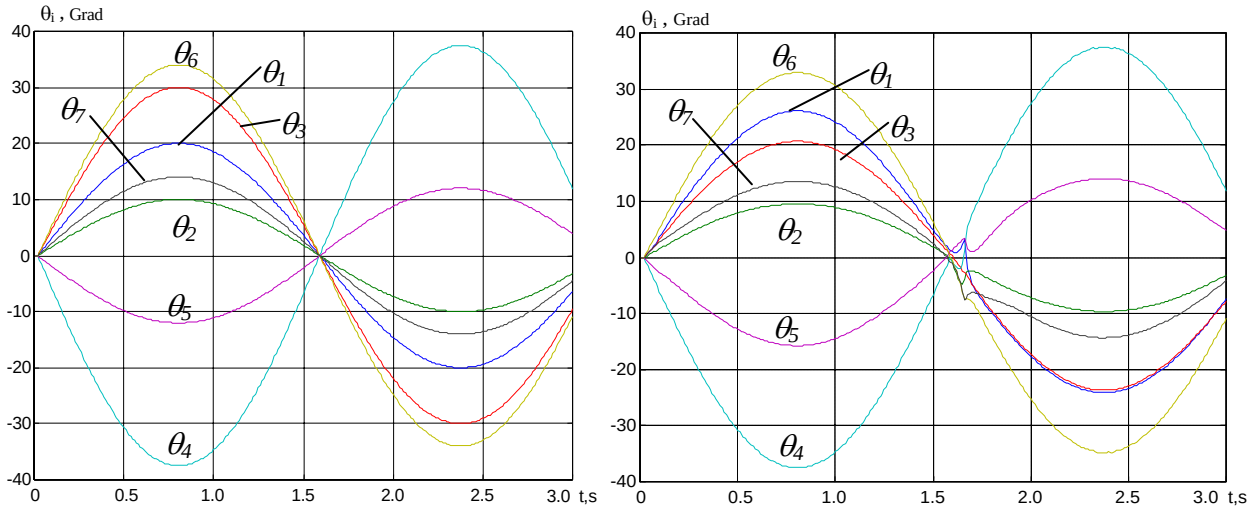


Abbildung 5.15: Sinusförmige Trajektorien der Gelenkwinkel (links) und der Verlauf der Gelenkwinkel im Ausgang der inversen Kinematik (rechts)

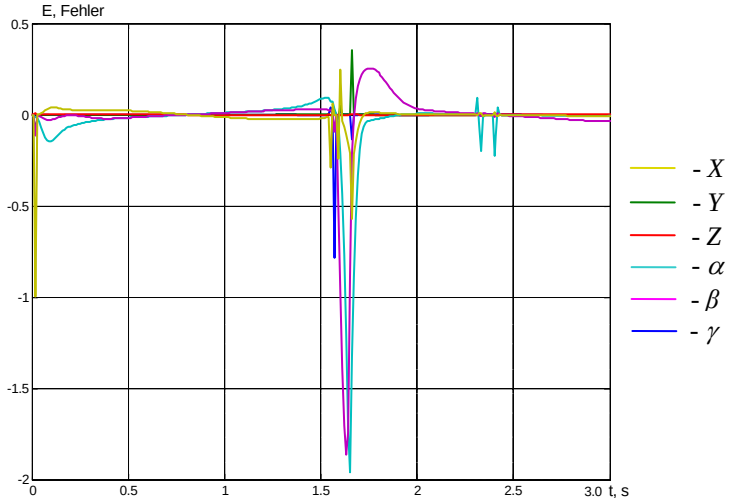


Abbildung 5.16: Die Fehlerfunktion der Position und Orientierung des Endeffektors in einem singulären Punkt

Nach der erfolgreichen simulativen Erprobung wurden die Programme zur Berechnung der Kinematik in der Gesamtsoftware des Roboters (siehe Anhang A) implementiert, um experimentell ebenfalls die Effektivität des vorgeschlagenen Verfahrens zu untersuchen und nachzuweisen. Dafür wurde, wie auch bei den simulativen Untersuchungen, eine dreieckförmige Trajektorie im kartesischen Arbeitsraum des Roboters generiert, um die Summe aller quadratischen Fehlerfunktionen bzgl. der Position und Orientierung des Endeffektors (E) aus experimentellen Daten zu ermitteln.

Im Gegensatz zum Signalflussbild der simulativen Untersuchung (siehe Abb. 5.12) wird am Ausgang der inversen Kinematik der 7-achsige Roboter *amtec r7* eingeschlossen. Die mit Hilfe der Inkrementalgeber gemessenen Gelenkwinkel des Roboters werden dann mittels der direkten Kinematik in die Position und Orientierung des Endeffektors transformiert und zur Darstellung gebracht. Die Abbildung 5.18 zeigt die Struktur solcher Robotersteuerung.

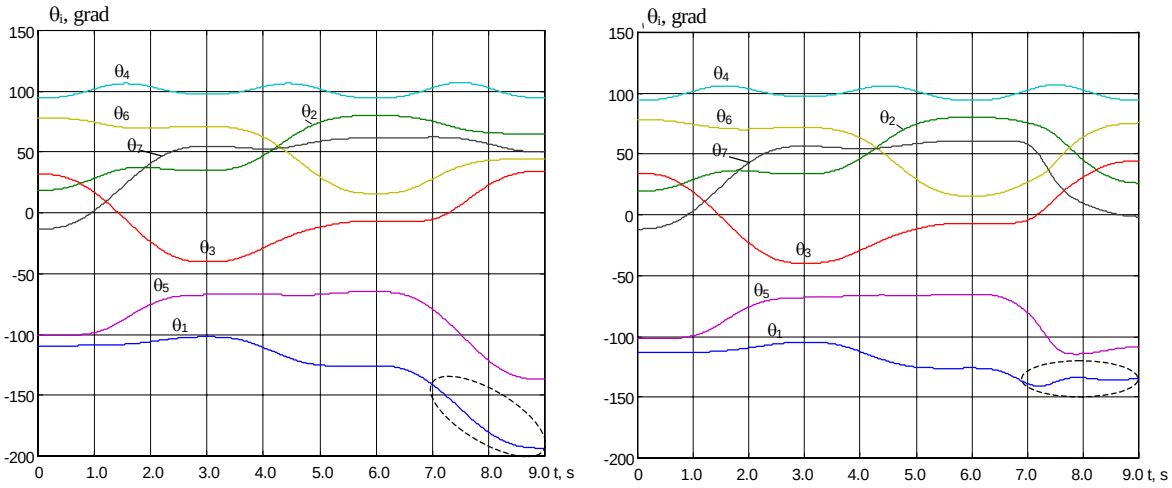


Abbildung 5.17: Konfiguration (Gelenkwinkel) des Roboters ohne (links) und mit (rechts) einer Überwachungsfunktion

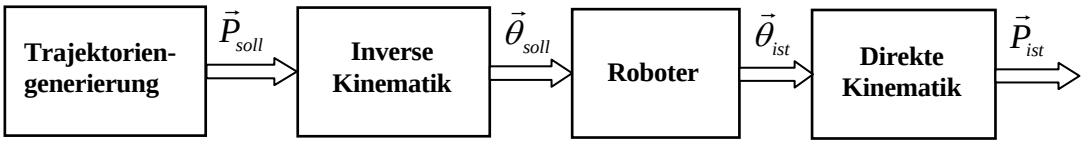


Abbildung 5.18: Struktur der Robotersteuerung

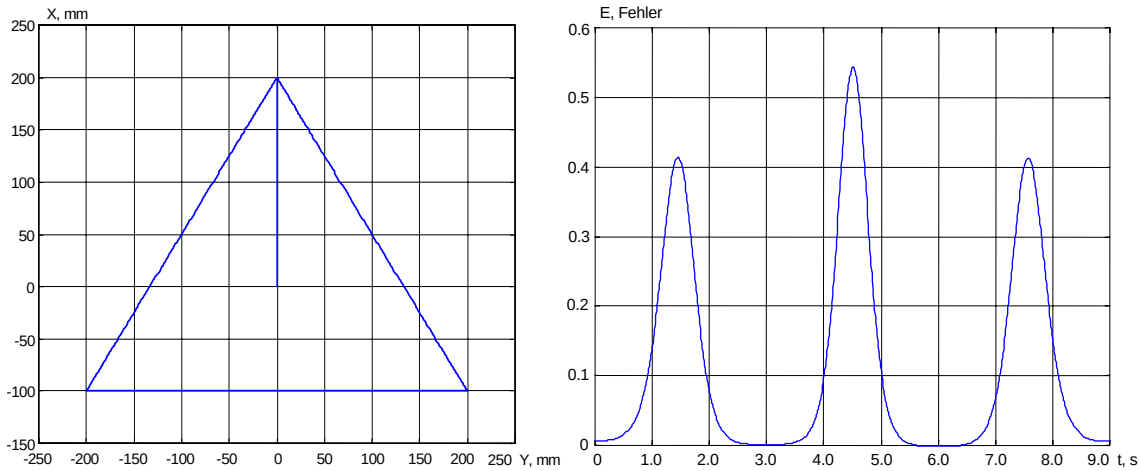


Abbildung 5.19: Planare dreieckförmige Trajektorie im Raum (links) und entsprechend die quadratischen Fehlerfunktion der Lage des Endeffektors (rechts)

Im Gegensatz zu den Ergebnissen der rein simulativen Untersuchung (siehe Abb. 5.13) wurde der quadratische Fehler zwischen den Soll- und Isttrajektorien im kartesischen Arbeitsraum deutlich größer (siehe Abb. 5.19), da die Dynamik des Roboters und der Messfehler der Inkrementalgeber ihre Auswirkung auf die Genauigkeit der Positionierung des Endeffektors im Verlauf der Trajektorie haben. Die qualitativen Zusammenhänge aus den simulativen Untersuchungen wurden jedoch uneingeschränkt bestätigt.

Kapitel 6

Kameramodell

Um mit Hilfe der PSD-Sensoren quantitative Aussagen über die Umgebung eines Roboters treffen zu können, ist eine ausreichend genaue Kameramodellbestimmung unerlässlich. Dies setzt eine ausreichend genaue Kamerakalibrierung voraus, im Zuge derer sowohl die Abbildungseigenschaften der PSD-Sensoren (innere Kameraparameter) als auch die Lage der Kamera bzgl. des Basiskoordinatensystems (äußere Kameraparameter) bestimmt werden müssen. Neben ausreichender Genauigkeit muss sich ein solches Verfahren insbesondere durch leichte Handhabbarkeit auszeichnen, so dass eine Kalibrierung am Einsatzort des Roboters erfolgen kann.

6.1 Grundlagen

In diesem Abschnitt des Kapitels wird die grundlegende Modellvorstellung einer Kamera beschrieben, auf die in den nächsten Abschnitten Bezug genommen wird. Eine Lochkamera ist die einfachste denkbare Kamera. Sie besteht aus einem allseitig umschlossenen dunklen Raum, der ein kleines Loch enthält, durch das Licht einfällt. Dieses Licht erzeugt ein Bild der Umgebung auf der Seitenfläche, die dem Loch gegenüber liegt. Bei der abstrakten Lochkamera, die hier beschrieben wird, bezeichnet man das Loch als Projektionszentrum und die Wand, auf die das Licht fällt, wird zur Projektionsebene. Beim davon abgeleiteten Mattscheibenmodell befindet sich die Bildebene vor dem Projektionszentrum (s. Abb. 6.1).

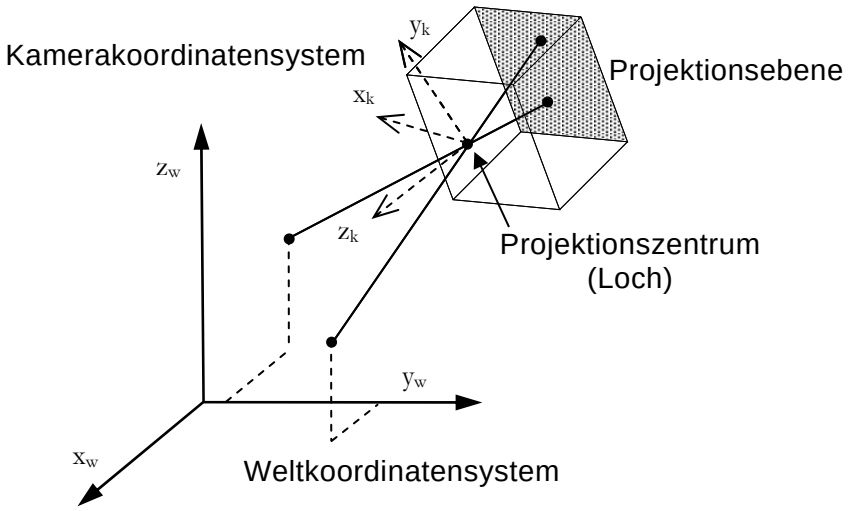


Abbildung 6.1: Lochkameramodell: Punkte aus dem Weltkoordinatensystem werden auf die Projektionsebene abgebildet, die parallel zur xy - Ebene des Systems ist

Die Lochkamera besitzt ein körperfestes Koordinatensystem, das sogenannte Kamerakoordinatensystem. Die optische Achse ist z.B. die z – Achse des Kamerasystems. Dann ist die Projektionsebene parallel zur xy – Ebene des Systems (siehe Abb. 6.1). Diese befindet sich im Abstand f vor oder hinter dem Projektionszentrum. Mathematisch wird die Transformation einer Szene der Umgebung auf die Projektionsebene der Lochkamera durch eine Koordinatentransformation der Szene vom Weltkoordinatensystem in das Kamerakoordinatensystem mit anschließender perspektivischer Projektion auf die Projektionsebene modelliert. Die Koordinaten eines Punkts im Kamerakoordinatensystem werden im folgenden durch $(x_k, y_k, z_k)^T$ bezeichnet. Die Sensorkoordinaten eines Punktes werden durch (x_s, y_s) bezeichnet. Die Weltkoordinaten sind $(x_w, y_w, z_w)^T$. Das Kamerakoordinatensystem kann eine beliebige Lage zum Weltkoordinatensystem einnehmen. Man erhält die Koordinaten von gegebenen Weltpunkten im Kamerakoordinatensystem durch die oben beschriebene Koordinatentransformation. Die Parameter dieser Transformation dienen zugleich als eindeutige Charakterisierung der Lage der Kamera im Weltkoordinatensystem. Eine solche beliebige invertierbare Koordinatentransformation besteht aus einer Translation und einer Rotation, die hintereinander ausgeführt werden. Dabei ist die Reihenfolge der Transformationen wichtig. Die beiden Operationen sind also nicht kommutativ. Damit gilt:

$$\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = R \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} = R \cdot p_w + t, \tag{6.1}$$

wobei R eine Rotationsmatrix und t ein Translationsvektor sind. R und t beschreiben die Position und Orientierung der Kamera im Raum. Die Rotationsmatrix kann dabei, wie im Kapitel 5 erläutert wurde, durch Gleichung (5.9) beschrieben werden:

$$R = \begin{bmatrix} \cos\gamma\cos\beta & \cos\gamma\sin\beta\sin\alpha - \sin\gamma\cos\alpha & \cos\alpha\sin\beta\cos\gamma + \sin\alpha\sin\gamma \\ \sin\gamma\cos\beta & \sin\gamma\sin\beta\sin\alpha + \cos\gamma\cos\alpha & \cos\alpha\sin\beta\sin\gamma - \sin\alpha\cos\gamma \\ -\sin\beta & \cos\beta\sin\alpha & \cos\beta\cos\alpha \end{bmatrix} \tag{6.2}$$

Das Kamerakoordinatensystem ist so gewählt, dass sein Ursprung im optischen Zentrum liegt und seine z-Achse mit der optischen Achse zusammenfällt.
Es ist bei der Betrachtung von Koordinatentransformationen zweckmäßig, die Gleichung (6.1) in homogenen Koordinaten darzustellen, da diese eine einheitliche Beschreibung von Rotation und Translation erlauben.
In der Computergrafik werden homogene Koordinaten als Standardkoordinatensystem benutzt. Ein 3D-Punkt $(x, y, z)^T$ wird in homogenen Koordinaten zu $(x, y, z, 1)^T$. Der Vorteil der homogenen Koordinaten ist, dass alle Transformationen (Verschiebung, Rotation, Perspektivische Projektion) durch eine 4x4 Matrix repräsentiert werden können. Durch die Einführung von homogenen Koordinaten lässt sich die Gleichung (6.1) in Matrixschreibweise wie folgt darstellen:

$$\begin{bmatrix} x_k \\ y_k \\ z_k \\ 1 \end{bmatrix} = \begin{bmatrix} & R_{3x3} & & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \tag{6.3}$$

bzw. in Kurzform

$$p_k = T \cdot p_w. \tag{6.4}$$

Eine Kameraaufnahme ist die Projektion einer 3-dimensionalen Szene auf eine 2-dimensionale Sensorfläche. Um eine Beziehung zwischen Szenenpunkt und Bildpunkt zu bekommen, wird die Projektion mit einem idealisierten Kameramodell nachgebildet. Durch eine Transformation der Koordinatensysteme kann man zwischen den verschiedenen Darstellungsformen derselben Szene wechseln. Um die Projektion geometrisch beschreiben zu können benötigen wir einige Kameraparameter, wie z.B. Brennweite oder Position der Kamera im Raum, welche durch die damit begründete Kalibrierung der Kamera gewonnen werden.

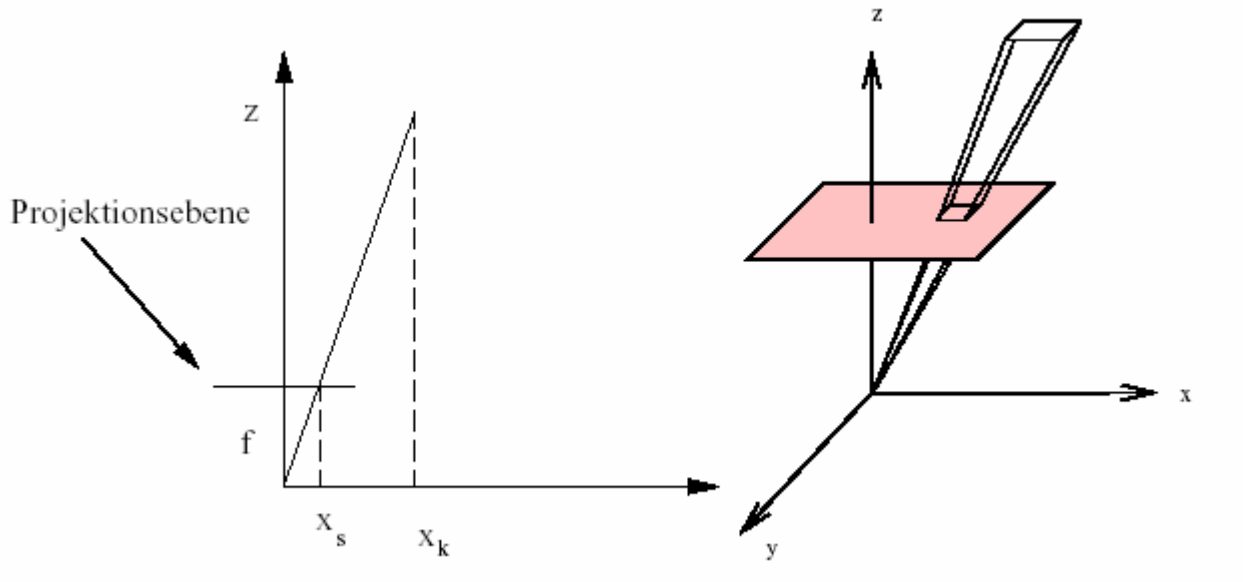


Abbildung 6.2: Perspektivische Projektion

Die Entstehung des Bildes in der Kamera kann durch eine Zentralprojektion beschrieben werden. Wie man in Abbildung 6.2 sehen kann, lassen sich die Koordinaten des Sensorpunkts, auf den ein Punkt im Kamerakoordinatensystem abgebildet wird, gemäß den Strahlensätzen berechnen. Werden Verzeichnungen durch Linsenfehler vernachlässigt, so ist jeder Weltpunkt $P(x_w, y_w, z_w)$ mit seinem korrespondierenden Bildpunkt $i(x_s, y_s)$ durch eine gerade Linie verbunden, die durch den Brennpunkt des Linsensystems der Kamera verläuft.
Damit lassen sich die Bildkoordinaten eines Punkts im Kamera-Koordinatensystem aus den folgenden perspektivischen Gleichungen herleiten:

$$\frac{x_s}{f} = \frac{x_k}{z_k} \rightarrow x_s = f\,\frac{x_k}{z_k}. \tag{6.5}$$

Analog gilt auch:

$$\frac{y_s}{f} = \frac{y_k}{z_k} \rightarrow y_s = f\,\frac{y_k}{z_k}. \tag{6.6}$$

f ist hierbei die Brennweite (Abstand zwischen Projektionszentrum K und Bildhauptpunkt H). Bildhauptpunkt H ist der Punkt auf der Bildebene, in dem die optische Achse die Sensorebene schneidet. Abbildung 6.3 erläutert die verwendeten Parameter anhand des Mattscheibenmodells und begründet die Gültigkeit der Gleichungen (6.5) und (6.6).

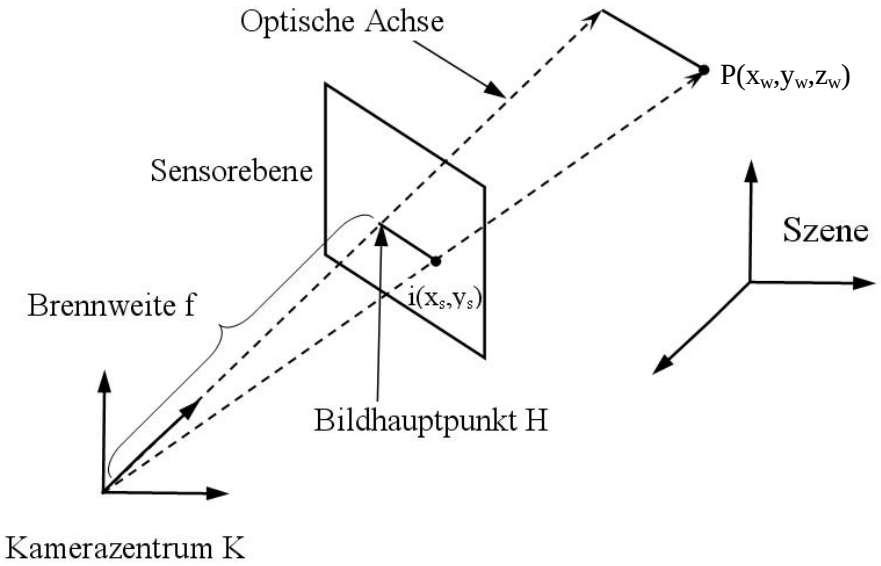


Abbildung 6.3: Die Parameter des Mattscheibenmodells

6.2 Bildaufnahmesystem

Im vorigen Abschnitt wurde die Modellvorstellung einer Kamera auf das Beispiel eines Lochkameramodells (Mattscheibenmodell) bezogen. In der Realität hat man es nicht nur mit einer derartigen Kamera zu tun, sondern mit einem ganzen System verschiedener Komponenten, die eine Szene der realen Welt in ein digitales Bild überführen. Ein solches System wird im folgenden als *Bildaufnahmesystem* bezeichnet. In diesem Abschnitt werden die einzelnen Komponenten dieses Systems beschrieben. Die Abbildung 6.4 zeigt ein Bildaufnahmesystem, dessen Struktur mit der im Robotersteuerungssystem eingesetzten übereinstimmt. Es enthält folgende Komponenten:

- Linsensystem: bildet eine Szene auf dem Sensor ab
- PSD- Chip: wandelt die Helligkeitsinformation in elektrische Information um
- Verstärker: verstärkt das analoge Sensorsignal
- A/D-Wandler: digitalisiert das analoge Videosignal

Abbildung 6.4: Komponenten eines Bildaufnahmesystems

6.2.1 Linsensystem

Das Linsensystem hat die Aufgabe, ein Abbild des Gegenstandraums auf der Bildebene zu erzeugen. Will man mit der Kamera genaue Messungen durchführen, muss man nichtlineare Effekte berücksichtigen, die von der Abweichung der Wirkung des Linsensystems vom Ideal

des Lochkameramodells herrühren. Die realen Linsensysteme haben folgende wesentlichen Abbildungsfehler:

- **Verzerrung**
- **Dezentrierung**

Die Ursache der Verzerrung liegt darin, dass die Transversalvergrößerung M_T eine mehr oder weniger bedeutsame Funktion des außeraxialen Abstandes y_i ist [6]. Der Abstand wird sich daher von dem unterscheiden, der von der Theorie achsnaher Strahlen vorhergesagt wird. Dieser Fehler macht sich durch eine Deformation des Bildes als ganzes bemerkbar (siehe Abb. 6.5), wobei dennoch jeder Punkt scharf abgebildet wird. Jeder Bildpunkt wird radial vom Mittelpunkt nach außen verschoben, wobei diese Verschiebung mit der Größe des Abstandes vom Mittelpunkt wächst. Die Linsenfehler beeinflussen die Positionen von Details des Bildes, nicht aber den Informationsgehalt des Bildes. Dies gilt zumindest theoretisch, wenn die Linsenfehler nicht zu groß sind.

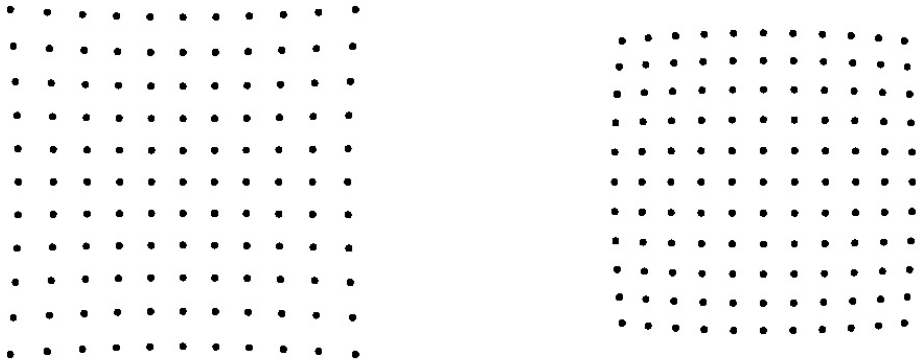
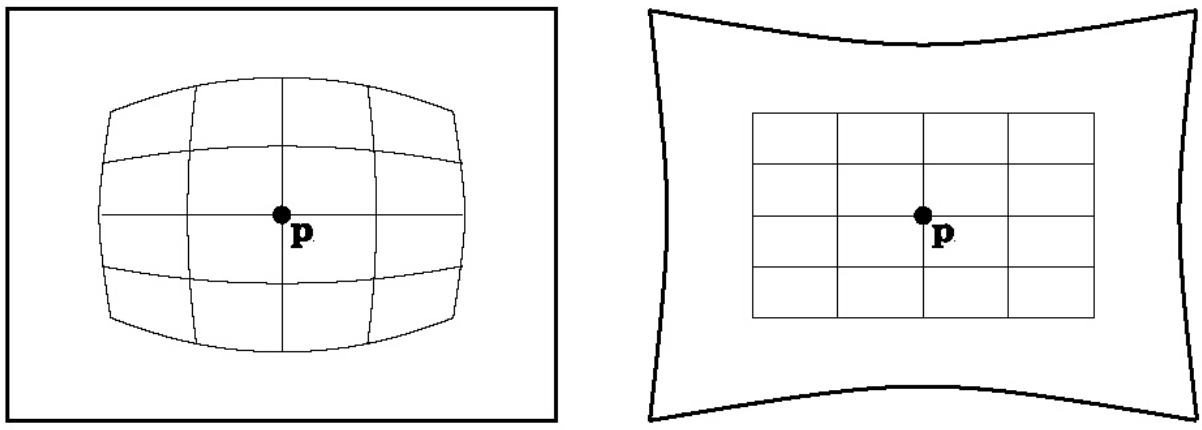


Abbildung 6.5: Auswirkung von positiver und negativer radialer Verzerrung

Die Abbildung 6.6(a) zeigt das Beispiel einer negativen radialen Verzerrung (tonnenförmige Verzerrung), die bei Kameras mit kurzer Brennweite häufig vorkommt. Die Geraden werden dabei nicht als Geraden abgebildet. Sind die Eigenschaften der Verzerrung bekannt, kann das Bild mit einer inversen Transformation entzerrt werden, so dass zwischen den Weltkoordinaten und dem entzerrten Bild wieder ein lineares Kameramodell (Lochkameramodell) verwendet werden kann (siehe Abb. 6.6(b)).



(a) Tonnenförmige Verzerrung (b) Entzerrtes Bild

Abbildung 6.6: Linsenverzerrung

Die mathematische Beschreibung der Linsenverzerrung wird im folgenden Abschnitt 6.3 genauer dargestellt. Eine Linse hat zwei Symmetrieachsen, eine optische und eine mechanische Achse. Die optische Achse ist die Linie, die die Krümmungszentren der beiden Oberflächen der Linse verbindet. Die mechanische Achse wird durch den abgeschliffenen Rand der Linse festgelegt. Idealerweise fallen die beiden Achsen zusammen – praktisch nicht. Die Differenz wird Dezentrierung genannt (siehe Abb. 6.7).

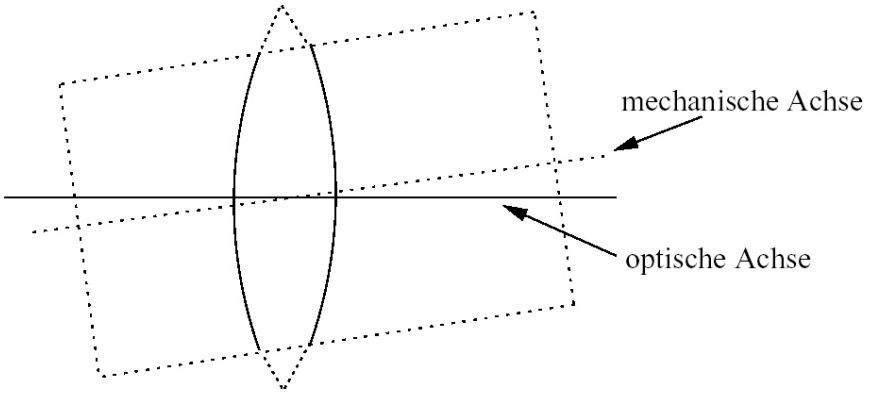


Abbildung 6.7: Die Dezentrierung einer Linse

6.2.2 PSD-Chip, Verstärker und A/D-Wandler

Wie schon im Kapitel 3 beschrieben wurde, werden wegen ihrer hohen Genauigkeit, ihrer Schnelligkeit und ihrer geringen Nichtlinearität die positionsempfindlichen Detektoren (PSD) in den Kameras installiert. Die modulierten Leuchtpunkte der Laserdioden werden auf dem PSD- Chip abgebildet und die Helligkeitsinformation in elektrische Signale der Kamera (X, Y) umgewandelt. Der Positionsverstärker OT-301 verstärkt die PSD- Ausgangssignale für die Achsen X und Y, und bildet ein jeweiliges Summensignal (S). Dann werden diese analogen Signale mittels des A/D- Wandlers digitalisiert und an das dSpace- Signalprozessorsystem zur weiteren Verarbeitung geschickt.
In der Realität treten Störungen bei der Messung der Signale der Laserdioden vor allem wegen der ungleichmäßigen Grundhelligkeit des Arbeitsraums, der Reflexion des Lichtes der Laserdioden an Roboteroberflächen und der Wirkung des Restlichtes der Laserdioden auf.
Um die durch ungleichmäßige Grundhelligkeit des Arbeitsraums auftretenden Störungen zu eliminieren, sind die dafür notwendigen Näherungsformeln zu bestimmen. Die Abbildung 6.8 zeigt das nach einer Pause auftretende Signal der Laserdiode bzgl. der *x*- Achse. Diese Pause dient zur Messung des Summensignals des Grundlichtes (S_0).
Es sind also definitionsgemäß gegeben:

– Summensignal:

$$S = \int L(x)dx \tag{6.7}$$

– Schwerpunktskoordinate:

$$x = \frac{\int L(x) \cdot xdx}{\int L(x)dx} \tag{6.8}$$

– $$\int L(x) \cdot xdx = x \cdot \int L(x)dx = x \cdot S \tag{6.9}$$

Messbar sind folgende Signale:

– Grundlicht

$$S_0 = \int L_0(x)dx, \qquad x_0 = \frac{\int L_0(x)\cdot x dx}{\int L_0(x)dx} \tag{6.10}$$

– Laserdioden-Licht mit Grundlicht

$$S_1 = \int L_1(x)dx, \qquad x_1 = \frac{\int L_1(x)\cdot x dx}{\int L_1(x)dx}, \qquad L_1(x) = L_0(x) + L_{LED}(x) \tag{6.11}$$

Die Schwerpunktskoordinate der Laserdiode L_{LED} ohne Einfluss des Grundlichtes wird definitionsgemäß durch folgende Gleichung bestimmt:

$$x_{LED} = \frac{\int L_{LED}(x)\cdot x dx}{\int L_{LED}(x)dx} \tag{6.12}$$

oder

$$x_{LED} = \frac{\int L_{LED}(x)\cdot x dx}{\int L_{LED}(x)dx} = \frac{\int (L_1(x) - L_0(x))\cdot x dx}{\int L_1(x)dx - L_0(x)dx} = \frac{x_1 S_1 - x_0 S_0}{S_1 - S_0} \tag{6.13}$$

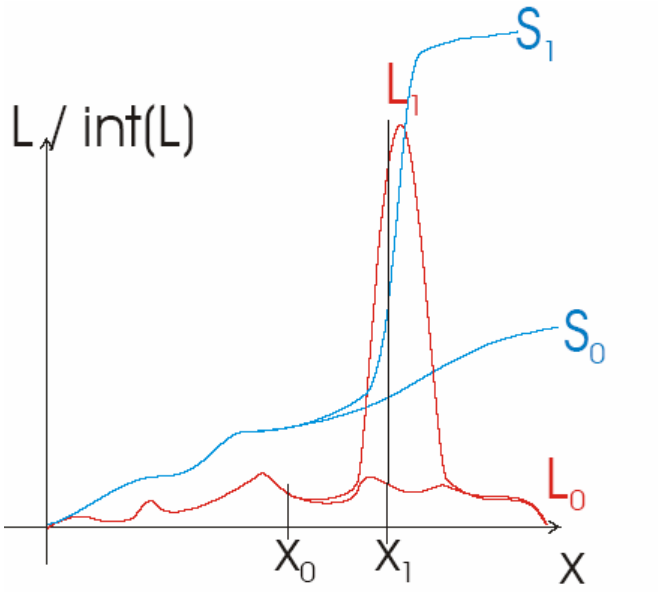


Abbildung 6.8: Die voraussichtliche Funktion des Laserdiodensignals mit dem Einfluss der Grundhelligkeit

Analog zu Gl. (6.13) gilt auch:

$$y_{LED1} = \frac{S_1 y_1 - S_0 y_0}{S_1 - S_0} \tag{6.14}$$

Damit gilt allgemein:

$$x_{LEDi} = \frac{S_i x_i - S_0 x_0}{S_i - S_0}$$
$$y_{LEDi} = \frac{S_i y_i - S_0 y_0}{S_i - S_0}, \quad (6.15)$$

wo *i=1…4* die Zahl der jeweiligen Dioden ist.
Durch die Gleichungen (6.15) wird der Einfluss des Grundlichtes näherungsweise eliminiert. Die anderen bereits oben beschriebenen Störungen wurden in dieser Arbeit nicht berücksichtigt.

6.3 Kamerakalibrierung

Die Kalibrierung dient dazu, die Abbildungseigenschaften der Kameras zu bestimmen. Sie stellt die Verbindung zwischen den Punkten der dargestellten Objekte in der realen dreidimensionalen Welt und ihren Projektionen in den zweidimensionalen Bildern her (siehe Abb. 6.1). Dabei gilt es, die inneren und äußeren Kameraparameter, die sich aus einem vorab (s. Abschnitt 6.1) gewählten Kameramodell heraus ergeben, mit möglichst hoher Genauigkeit unter vertretbarem Aufwand zu bestimmen.
Die inneren Kameraparameter sind invariant gegenüber der Verschiebung und Rotation der Kamera im Raum (nicht aber gegen Austausch des Sensors oder des Objektivs). Mit den inneren Parametern werden die Abbildungseigenschaften der Kamera beschrieben. Hierzu gehören z.B. die Brennweite, die Bildebenenverkippung, die Lage des Hauptpunktes auf der Bildebene und darüber hinaus Parameter, die eine Verzeichnung der Linse als Abweichung vom optischen Ideal einer Lochkamera beschreiben. Erst mit der genauen Kenntnis dieser Parameter ist es möglich, die Fehler bei der Bildmessung (Bestimmung eines Punktes im Bild) zu eliminieren.
Die äußeren Parameter beschreiben die Position und Orientierung der Kameras in einem Bezugskoordinatensystem (hier: Objektkoordinatensystem). Diese Parameter verändern sich, wenn man die Kamera im Raum bewegt. Um die äußeren Parameter bestimmen zu können, müssen die inneren Parameter bereits bekannt sein.
Die ersten Algorithmen für die Kamerakalibrierung wurden für die Anwendung in der Photogrammetrie entwickelt. Diese Disziplin befasst sich mit der Vermessung von Strukturen fotografisch aufgenommener Bilder, wie z. B. Luft- oder Satellitenaufnahmen eines Geländes oder mikroskopische Fotografien von Werkstoffen oder biologischen Präparaten. Die Algorithmen lassen sich nach ihrer Komplexität und nach dem Grad der Interaktion, die für die Bestimmung der Kameraparameter notwendig sind, klassifizieren. Sie sind in [52, 53, 54, 80, 90] beschrieben. Daraus ist ersichtlich, dass ein einsetzbarer Kamerakalibrierungsalgorithmus zumindest folgenden Kriterien erfüllen sollte:

- **Autonomie**: Es sollte kein manuelles Bestimmen der Parameter notwendig sein.
- **Exaktheit**: Die Kalibrierung sollte möglichst genau sein und es sollte eine Aussage über die Größe des Kalibrierungsfehlers möglich sein.

- **Effizienz**: Die Berechnung der Kalibrierung muss den von der Anwendung vorgegebenen Zeitbedingungen genügen. Die Kalibrierung sollte daher in den meisten Fällen keine höherdimensionale nichtlineare Optimierung benötigen.
- **Anwendbarkeit**: Es sollten beliebige Standardkameras kalibrierbar sein.

Hinsichtlich ihrer Komplexität lassen sich die Verfahren in drei Klassen einteilen:

- **Vollständig nichtlineare Verfahren** ermöglichen eine Anpassung an beliebig genaue Kameramodelle. Notwendig ist dabei allerdings fast immer eine rechenintensive nichtlineare iterative Suche. Für die gesicherte Konvergenz und Stabilität ist außerdem die Wahl guter initialer Startbedingungen eine nicht immer leicht zu erfüllende Voraussetzung.
- **Lineare Verfahren** benötigen keine nichtlineare Optimierung und ermöglichen daher kurze Rechenzeiten. Diese Verfahren sind immer dann geeignet, wenn in Echtzeit kalibriert werden soll und keine hohen Anforderungen an die Kalibrierungsgenauigkeit gestellt werden müssen. Durch die häufig hohe Zahl an Unbekannten im linearen Gleichungssystem ist in verrauschter Umgebung nur eine begrenzte Genauigkeit zu erreichen. Außerdem können nichtlineare Parameter (z. B. Linsenverzerrungen) nicht modelliert werden.
- **Kombinierte Verfahren** setzen lineare Optimierung für die Berechnung eines Teils der Kameraparameter ein und berechnen die übrigen Parameter nichtlinear. Durch eine geschickte Aufteilung lässt sich die Rechenzeit für die nichtlinear optimierten Parameter minimieren und günstige Startwerte für die Iteration berechnen. Die Genauigkeit der Kalibrierung liegt in der Regel höher als bei rein linearen Verfahren.

Wie bereits im Abschnitt 6.2 beschrieben wurde, treten bei Einsatz konventioneller Kameras Abweichungen im Vergleich zur rein projektiven Abbildung auf. Ohne Berücksichtigung dieser Abweichungen werden die Bildinformationen verschlechtert. Da die Hauptabweichungen in den meisten Fällen mathematisch zu beschreiben sind, werden sie in der Regel im Kameramodell berücksichtigt. Für dieses System wird folgendes berücksichtigt:

1. Bildhauptpunktveschiebung

Für die dreidimensionalen Messungen ist es wichtig, die absoluten Sensorkoordinaten zu kennen. Aber der Hauptpunkt des Sensors fällt nicht mehr mit dem Hauptpunkt des Bildkoordinatensystems zusammen. Darum ist es wichtig, die Lage des Bildhauptpunktes zu ermitteln. Für die Kameras hängt die Lage des Bildhauptpunktes von vielen Faktoren, wie der Einbaulage des Sensorchips, der Lage des Objektivs, der Asymmetrie des Objektivs ab. Dies führt zur Verschiebung des Bildhauptpunktes vom Kamerakoordinatensystem. Der Bildhauptpunkt H ist ein Punkt auf der Bildebene, in dem die optische Achse die Sensorebene schneidet (s. Abb. 6.9).
Der Zusammenhang zwischen den Bild- und Sensorkoordinaten lässt sich durch folgende Gleichungen beschreiben:

$$\begin{aligned} x_s &= x_f + c_x \\ y_s &= y_f + c_y \end{aligned}, \tag{6.16}$$

wo (x_s, y_s) - die Bildkoordinaten, (x_f, y_f) - die Sensorkoordinaten und (c_x, c_y) - die Verschiebung zwischen Bild- und Sensorkoordinaten darstellen.

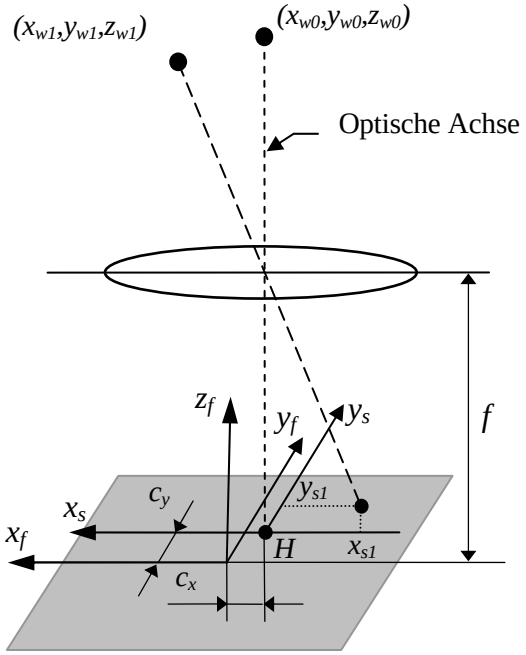


Abbildung 6.9: Sensor- und Bildkoordinatensystem

2. Radiale Linsenverzerrung

Wie bereits im Abschnitt 6.2 erwähnt wurde, weisen Abbildungen mit Linsen gewisse Fehler auf. Der Fehler, der sich am stärksten bemerkbar macht, ist die radiale Linsenverzerrung. Die radiale Linsenverzerrung wird durch unperfekte Herstellung von Linsen verursacht. Die Linsenverzerrungen einfacher Objektive resultieren daraus, dass ein durch die Blende begrenztes und in die Linse einfallendes Strahlenbündel mit zunehmendem Einfallswinkel durch die Linsenrandbereiche auf die Bildebene projiziert wird. Die Linsenverzerrung wird in der Regel als radial konstant angenommen. Das heißt sie kann als Funktion des Abstandes r des unverzerrten Punktes vom Ursprung des Bildkoordinatensystems modelliert werden:

$$P_d = f(r) \quad mit \quad r = \sqrt{x_s^2 + y_s^2} \tag{6.17}$$

Ein allgemeines Modell für die Linsenverzerrung ist die in [90] vorgeschlagene unendliche Reihe aus gradzahligen Potenzen. Ein idealer Sensorpunkt x_s wird durch folgende Gleichungen auf einen gestörten Sensorpunkt x_d abgebildet.

$$\begin{aligned} x_d &= x_s(1 + k_1 r^2 + k_2 r^4 + \ldots) \\ y_d &= y_s(1 + k_1 r^2 + k_2 r^4 + \ldots) \end{aligned} \tag{6.18}$$

In den meisten Fällen wird nur k_1 ermittelt, da der quadratische Term den bei weitem größten Teil der Verzerrung erfasst und die Anzahl der zu ermittelten Parameter so klein wie möglich gehalten werden sollte. Das Zentrum der radialen Verzerrung ist im Idealfall der Hauptpunkt. Die in [90] gemachte Aussage, dass die Genauigkeit nicht wesentlich leide, wenn der Bildmittelpunkt als Hauptpunkt fungiert, wird in [54] revidiert.

Um die Transformation der 3D-Weltkoordinaten eines Lichtpunktes zu 2D-Bildkoordinaten eines Kamerachips zu beschreiben, müssen die Linsenverzerrung und die Bildhauptpunktverschiebung berücksichtigt werden. Dafür müssen folgende 4 Schritten definiert werden:

1. Transformation Weltkoordinatensystem → Kamerakoordinatensystem
   $[(x_w, y_w, z_w) \rightarrow (x_k, y_k, z_k)]$

$$\begin{bmatrix} x_k \\ y_k \\ z_k \\ 1 \end{bmatrix} = \begin{bmatrix} & & & t_x \\ & R_{3x3} & & t_y \\ & & & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \tag{6.19}$$

   wobei R_{3x3} eine Rotationsmatrix und T ein Translationsvektor folgender Form sind:

$$R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix}, \qquad T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}. \tag{6.20}$$

   Gesucht sind die Kalibrierungsparameter *R* und *T*.

2. Transformation Kamerakoordinatensystem → Bildkoordinatensystem
   (ohne Berücksichtigung der Linsenverzerrung)

$$x_f = f \frac{x_k}{z_k} \quad und \quad y_f = f \frac{y_k}{z_k}. \tag{6.21}$$

   Wie schon im Abschnitt 6.1 beschrieben wurde, definiert die Gleichung (6.21) eine einfache perspektivische Projektion (Abb. 6.3).
   Gesucht ist der Kalibrierungsparameter *f* (Brennweite).

3. Berücksichtigung der Linsenverzerrung
   $[(x_f, y_f) \rightarrow (x_d, y_d)]$

$$\begin{aligned} x_f &= x_d (1 + k_1 r^2 + k_2 r^4) \\ y_f &= y_d (1 + k_1 r^2 + k_2 r^4) \\ r &= \sqrt{x_d^2 + y_d^2} \end{aligned} \tag{6.22}$$

   Wie bereits oben erwähnt wurde, sind theoretisch unendlich viele Verzerrungsparameter k_i (s. Gl. 6.18) zu betrachten. In der Praxis sind jedoch ein oder zwei Terme ausreichend.
   Gesucht sind die Kalibrierungsparameter k_1, k_2.

4. Transformation Sensorkoordinatensystem → Bildkoordinatensystem
   $[(x_d, y_d) \rightarrow (x_s, y_s)]$

$$\begin{aligned} x_s &= x_d + c_x \\ y_s &= y_d + c_y \end{aligned}, \tag{6.23}$$

   wobei c_x, c_y die Hauptpunktverschiebung entlang der *x* - und *y* - Achse.
   Gesucht sind die Kalibrierungsparameter c_x, c_y.

Dabei müssen folgende Parameter, die zu den inneren Kameraparametern gehören, bestimmt werden: f, k_1, k_2, c_x, c_y. Da die absoluten Ortkoordinaten des Kamerakoordinatensystems (äußere Kameraparameter) t_x, t_y, t_z anfänglich unbekannt sind, müssen sie ebenfalls ermittelt werden.

6.3.1 Ermittlung der inneren Kameraparameter

Da nicht alle Kameraparameter durch einen Optimierungsverfahren gleichzeitig bestimmt werden können, was später in diesem Abschnitt gezeigt wird, werden die inneren Parameter der Kamera zuerst durch eine 2D-Kalibrierung ermittelt. In diesem Zusammenhang wird unter 2D in [80] verstanden, dass die Kamera senkrecht zum Endeffektor steht, dass also keine perspektivischen Effekte auftreten, da die Achsen der Bildebene bereits richtungsgleich zur absoluten x- und y-Achse liegen.

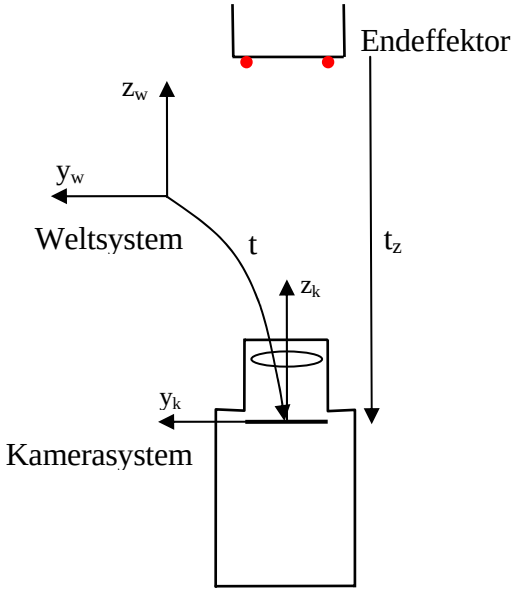


Abbildung 6.10: Aufbau zur 2D-Kalibrierung der inneren Parameter

Für den obigen Aufbau (s. Abb. 6.10) kann die Kameratransformation durch eine Rotation um die z-Achse, eine Verzerrung und eine Verschiebung beschrieben werden. Die Abbildungstransformation für die 2D-Kalibrierung lässt sich also in folgenden Schritten beschreiben (Abb. 6.10):

1. Transformation Weltkoordinatensystem → Kamerakoordinatensystem:

   Da die Blickrichtung der Kamera senkrecht zum Endeffektor (Weltkoordinatensystem) ist, haben beide Systeme identische z-Achsen. Die Transformation in Kamerakoordinaten kann daher wie folgt als eine Rotation um die z-Achse mit anschließender Verzerrung und Verschiebung beschrieben werden (s. Gl. 6.2, 6.3, 6.4):

   $$p_k = T \cdot p_w$$

   oder:

$$\begin{bmatrix} x_k \\ y_k \\ z_k \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 & t_x \\ \sin\gamma & \cos\gamma & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \tag{6.24}$$

Danach müssen die bereits im Abschnitt 6.3 beschriebenen Transformationen 2-4 durchgeführt werden, die durch die Gleichungen (6.21-6.23) beschrieben sind.
Führt man danach alle Transformationen (Gl. 6.21-6.24) durch, mit Vernachlässigung der radialen Linsenverzerrung (Gl. 6.22), so ergibt sich folgender Zusammenhang zwischen Weltkoordinaten und Sensorkoordinaten.

$$\begin{aligned} x_s &= f\,\frac{x_w\cos\gamma - y_w\sin\gamma + t_x}{z_w + t_z} + c_x \\ y_s &= f\,\frac{x_w\sin\gamma + y_w\cos\gamma + t_y}{z_w + t_z} + c_y \end{aligned} \tag{6.25}$$

Aus der Gleichung (6.25) ist ersichtlich, dass nicht alle Parameter getrennt voneinander bestimmbar sind (z. B. mittels des Optimierungsverfahrens). So lässt sich bei diesem Problem nur der Quotient f/t_z bestimmen. Die Lösung dieses Systems von Gleichungen stellt in der Regel noch keine ideale Lösung dar, da zwischen den einzelnen Parametern Kopplungen bestehen. In [80] wurde eine Methode dargestellt, die es erlaubt, alle inneren Parameter getrennt voneinander zu bestimmen. Sie wird nachfolgend in dieser Arbeit beschrieben.
Um die inneren Parameter zu ermitteln, muss zuerst ein Leuchtpunkt der Laserdiode im Weltkoordinatensystem (x_{w0}, y_{w0}) bestimmt werden, der in der optischen Achse liegt und auf der Bildebene als Bildhauptpunkt (x_{s0}, y_{s0}) projiziert wird. Dabei ist es egal, wie groß der Abstand dieses Punktes von der Linse entlang der z-Achse wird (s. Abb. 6.11).

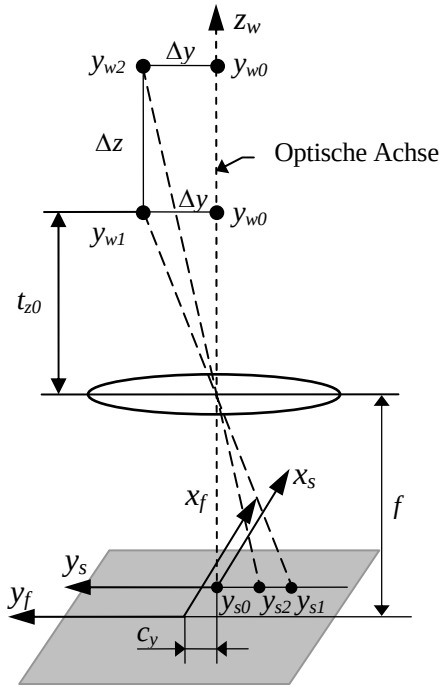


Abbildung 6.11: Bestimmung des Zentrums einer Linse

Gemäß der Verhältnistheorie kann man folgende Gleichungen notieren.
Für die y-Achse gilt:

$$\frac{y_{s1} - y_{s0}}{f} = \frac{y_{w1} - y_{w0}}{z_w + t_{z0}}$$
$$\frac{y_{s2} - y_{s0}}{f} = \frac{y_{w2} - y_{w0}}{z_w + t_{z0} + \Delta z} \tag{6.26}$$

Für x-Achse ergibt sich:

$$\frac{x_{s1} - x_{s0}}{f} = \frac{x_{w1} - x_{w0}}{z_w + t_{z0}}$$
$$\frac{x_{s2} - x_{s0}}{f} = \frac{x_{w2} - x_{w0}}{z_w + t_{z0} + \Delta z} \tag{6.27}$$

Aus der Gleichung (6.26) werden der Bildhauptpunkt und der entsprechende Punkt im Weltkoordinatensystem entlang der optischen Achse berechnet.
Für die y-Achse gilt:

$$y_{s0} = \frac{y_{s2}(z_w + t_{z0} + \Delta z) - y_{s1}(z_w + t_{z0})}{\Delta z}$$
$$y_{wo} = \frac{(z_w + t_{z0}) \cdot (y_{s0} - y_{s1}) + f \cdot y_{w1}}{f} \tag{6.28}$$

Für x-Achse gemäß der Gl. (6.27) ergibt sich:

$$x_{s0} = \frac{x_{s2}(z_w + t_{z0} + \Delta z) - x_{s1}(z_w + t_{z0})}{\Delta z}$$
$$x_{wo} = \frac{(z_w + t_{z0}) \cdot (x_{s0} - x_{s1}) + f \cdot x_{w1}}{f} \tag{6.29}$$

Durch Gleichungen (6.28-6.29) wird also ein Leuchtpunkt der Laserdiode im Weltkoordinatensystem (x_{w0}, y_{w0}), der auf der optischer Achse liegt, bestimmt.

Danach muss ein Hauptpunkt der Kamera bestimmt werden, was in zwei Schritten erfolgt. Zuerst werden die exakten Werte für f und für $t_{\tilde{z}}$ bestimmt. Dazu ist notwendig, senkrechte Projektionen des Endeffektors mit einer leuchtenden Diode in verschiedenen Distanzen mit bekanntem Relativabstand zu messen. Man kann dies durch einen experimentellen Aufbau, wie in Abb. 6.12 gezeigt wird, erreichen. Mit Kenntnis dieser Parameter kann dann in einem zweiten Schritt der Hauptpunkt (c_x, c_y) der Kamera bestimmt werden.
Zwei verschiedene Weltpunkte y_w^a und y_w^b werden auf verschiedene Sensorpunkte abgebildet. Gemäß der Gleichung (6.25) werden diese Punkte wie folgt beschrieben:

$$y_s^a = \frac{f \cdot (x_w^a \sin\gamma + y_w^a \cos\gamma + t_y)}{z_w + t_z} + c_y$$
$$y_s^b = \frac{f \cdot (x_w^b \sin\gamma + y_w^b \cos\gamma + t_y)}{z_w + t_z} + c_y \tag{6.30}$$

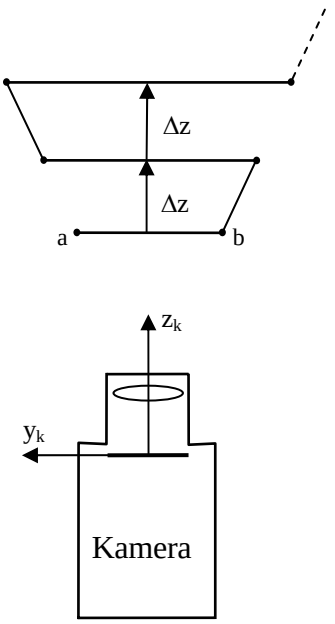


Abbildung 6.12: Messung der relativen Verschiebung

Die absoluten Sensorkoordinaten der beiden Punkte sind nicht bekannt, da der Hauptpunkt der Kamera unbekannt ist. Der relative Abstand zwischen zwei Punkten ist jedoch bekannt und hängt von absoluten Sensorkoordinaten der Punkte nicht ab.

$$\Delta y_s^{ab}(t_z) = y_s^a - y_s^b = \frac{f}{z_w + t_z}(y_w^a - y_w^b)\cos\gamma\,, \tag{6.31}$$

da $(x_w^a - x_w^b)$ =0 ist, weil sich die Position bzgl. der *x*-Achse nicht ändert.
Die Änderung von t_z ist bekannt und lässt sich so schreiben:

$$t_z(n) = t_z^0 + n\Delta z \tag{6.32}$$

Setzt man das wieder in die Gleichung (6.31) ein, so erhält man:

$$\Delta y_s^{ab}(n) = \frac{f}{z_w + t_z^0 + n\Delta z}\Delta y_w^{ab}(n)\cos\gamma \tag{6.33}$$

Die obige Gleichung kann man in folgende Form bringen:

$$\Delta y_s^{ab}(n)t_z^0 - f\cos\gamma\Delta y_w^{ab} = -\Delta y_s^{ab}(n)\cdot(z_w + n\Delta z) \tag{6.34}$$

Aus der obigen Gleichung sieht man, dass die Parameter *f* und γ nicht getrennt voneinander bestimmt werden können. So lässt sich nur das Produkt $f\cdot\cos\gamma$ bestimmen. Deswegen wird die Kamera experimentell so befestigt, dass $\gamma=0$ ist. So ergibt sich:

$$\Delta y_s^{ab}(n)t_z^0 - f\Delta y_w^{ab} = -\Delta y_s^{ab}(n)\cdot(z_w + n\Delta z) \tag{6.35}$$

Mit mindestens zwei relativen Beobachtungsabständen kann aus der Gl. (6.35) nun ein lineares Gleichungssystem aufgestellt und gelöst werden.

$$\begin{pmatrix} \Delta y_s^{ab}(0) & -\Delta y_w^{ab} \\ \vdots & \vdots \\ \Delta y_s^{ab}(n) & -\Delta y_w^{ab} \end{pmatrix} \cdot \begin{pmatrix} t_z^0 \\ f \end{pmatrix} = \begin{pmatrix} -\Delta y_s^{ab}(0) \cdot z_w \\ \vdots \\ -\Delta y_s^{ab}(n) \cdot (z_w + n\Delta z) \end{pmatrix} \tag{6.36}$$

Für drei relative Beobachtungsabständen ($n=0\ldots2$) wurde das lineare Gleichungssystem (6.36) in unserem Fall gelöst. Nun sind die Brennweite f und der absolute Abstand t_z der Linse bekannt (siehe Tabelle 6.1).

Damit kann nun der Hauptpunkt der Abbildung in den beiden Koordinatenrichtungen (c_x, c_y) ermittelt werden. Dazu wird nicht, wie eben, die Abhängigkeit der Distanz zwischen zwei auf den Sensor projizierten Weltpunkten betrachtet, sondern die Änderung der absoluten Position eines Punktes entlang einer optischen Achse (z-Achse).
Die Transformationsgleichungen lauten dabei:

$$\begin{aligned} x_s &= f\,\frac{x_w + t_x}{z_w + t_z} + c_x \\ y_s &= f\,\frac{y_w + t_y}{z_w + t_z} + c_y \end{aligned} \tag{6.37}$$

Die Änderung von t_z ist bekannt und lässt sich so schreiben:

$$t_z(n) = t_z^0 + n\Delta z \tag{6.38}$$

Der Wert für f ist ebenso bekannt. Damit können die Gleichungen (6.37) umgeschrieben werden in:

$$\begin{aligned} x_s \cdot (z_w + t_z^0 + n\Delta z) &= f \cdot x_w + f \cdot t_x + c_x \cdot (z_w + t_z^0 + n\Delta z) \\ y_s \cdot (z_w + t_z^0 + n\Delta z) &= f \cdot y_w + f \cdot t_y + c_y \cdot (z_w + t_z^0 + n\Delta z) \end{aligned} \tag{6.39}$$

Für n Beobachtungspunkte kann man daraus ein System von n linearen Gleichungen bilden und lösen:
für x:

$$\begin{pmatrix} f & z_w + t_z^0 \\ \vdots & \vdots \\ f & z_w + t_z^0 + n\Delta z \end{pmatrix} \cdot \begin{pmatrix} t_x \\ c_x \end{pmatrix} = \begin{pmatrix} x_s(0) \cdot (z_w + t_z^0) - f \cdot x_w(0) \\ \vdots \\ x_s(n) \cdot (z_w + t_z^0 + n\Delta z) - f \cdot x_w(n) \end{pmatrix} \tag{6.40}$$

für y:

$$\begin{pmatrix} f & z_w + t_z^0 \\ \vdots & \vdots \\ f & z_w + t_z^0 + n\Delta z \end{pmatrix} \cdot \begin{pmatrix} t_y \\ c_y \end{pmatrix} = \begin{pmatrix} y_s(0) \cdot (z_w + t_z^0) - f \cdot y_w(0) \\ \vdots \\ y_s(n) \cdot (z_w + t_z^0 + n\Delta z) - f \cdot y_w(n) \end{pmatrix} \tag{6.41}$$

Für 5 Beobachtungspunkte wurden die Gleichungssysteme (6.40-6.41) in unserem Fall gelöst und damit die absoluten Abstände t_x, t_y und die Hauptpunktverschiebung (c_x, c_y) der Kamera ermittelt. (siehe Tabelle 6.1).

Weiterhin ist die Linsenverzerrung der Kameras zu bestimmen, wobei alle anderen inneren Parameter schon bekannt sind. Dazu sind mehrere auf dem gesamten PSD- Chip projizierte Punkte notwendig, um die Verzerrungskoeffizienten der Kameralinsen mit großer Genauigkeit

zu ermitteln. Abbildung 6.13 stellt eine im kartesischen Arbeitsraum (XY- Ebene) Trajektorie dar, die vom einen festgelegten Punkt zu den anderen verläuft. In jedem Punkt der Trajektorie von *0* bis *9* werden die Positionen des Endeffektors des Roboters im Stillstand gemessen. Dabei werden zehn auf dem PSD- Chip projizierte Punkte gemessen, die dann verschiedene Abstände vom Punkt *0* aufweisen.
Aus der Gleichungen (6.21-6.23) ergibt sich ein folgendes Gleichungssystem:

$$\begin{aligned} f\frac{x_w}{z_w} &= x_d(1+k_1r^2+k_2r^4) \\ f\frac{y_w}{z_w} &= y_d(1+k_1r^2+k_2r^4) \end{aligned}, \tag{6.42}$$

wo:

$$\begin{aligned} r &= \sqrt{x_d^2+y_d^2} \\ x_d &= x_s + c_x \\ y_d &= y_s + c_y \end{aligned} \tag{6.43}$$

sind.
Gemäß der Gleichung (6.42) lässt sich folgende quadratische Fehlerfunktion angeben, die mit Hilfe des Levenberg- Marquardt Optimierungsverfahrens zu minimieren ist:

$$E = \sum_{i=0}^{9}\left(\left(f\frac{x_{wi}}{z_{wi}} - x_{di}(1+k_1r_i^2+k_2r_i^4)\right)^2 + \left(f\frac{y_{wi}}{z_{wi}} - y_{di}(1+k_1r_i^2+k_2r_i^4)\right)\right)^2. \tag{6.44}$$

Nach Minimierung der Fehlerfunktion (6.44) sind die Koeffizienten k_1 und k_2 der radialen Linsenverzerrung bekannt (siehe Tabelle 6.1).

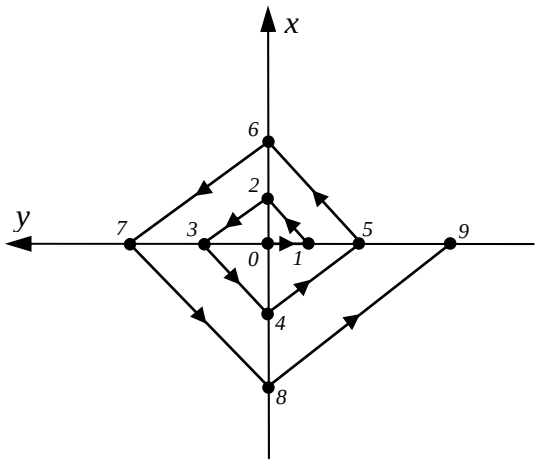


Abbildung 6.13: Trajektorie zur Bestimmung der radialen Linsenverzerrung

Für die Bestimmung der inneren Kameraparameter gemäß dem oben beschriebenen Verfahren wurde ein C- Programm „ikp.c“ geschrieben und erfolgreich für beide Kameras benutzt. Die Ergebnisse dieser Kameraparameterbestimmung unter Nutzung experimenteller Daten sind in Tabelle 6.1 dargestellt. Die inneren Kameraparameter f, k_1, k_2, c_x, c_y sind nicht positionsabhängig. Dazu wurden Experimente bei jeweils veränderten Kamerapositionen zur wiederholten Bestimmung dieser Parameter durchgeführt. Dabei betrugen die Abweichungen der ermittelten inneren Kameraparameter maximal 1.2%.

Parameter	Kamera 1	Kamera 2
Brennweite f	0.98	0.94
x - Koordinate Hauptpunkt c_x	-0.13	-0.20
y - Koordinate Hauptpunkt c_y	0.01	0.08
Linsenverzerrung k_1	$2.4*10^{-3}$	$2.8*10^{-3}$
Linsenverzerrung k_2	$1.0*10^{-5}$	$1.2*10^{-5}$
Verschiebung in x - Richtung t_x	405.0	405.0
Verschiebung in y - Richtung t_y	44,5	46,7
Verschiebung in z - Richtung t_z	70,2	69,7

Tabelle 6.1: Die experimentell bestimmten inneren Kameraparameter

Um diese inneren Kameraparameter weiter verwenden zu können, bedarf es einer quantitativen Abschätzung ihrer Genauigkeit im Vergleich zu realen Werten. Dies wurde mit den beiden PSD-Kameras mit senkrecht stehender z-Achse durchgeführt, nachdem deren Kameraparameter bestimmt wurden. Dann werden die Sensorwerte x_s und y_s der Kameras für einen Leuchtpunkt in ihrem Sichtbereich gemessen und die Weltkoordinaten dieses Punktes gemäß der Gleichungen (6.20-6.23) rückwärts berechnet. Dabei wird eine Differenz zwischen den mittels der direkten Kinematik ermittelten Positionen und den gemessenen Sensorwerten beider Kameras gebildet. Sie betrug maximal 0.11 mm, 0.13 mm und 0.21 mm bezüglich der Achsen X, Y und Z.

6.3.2 Ermittlung der äußeren Kameraparameter

Die Aufgabe der 3D-Kalibrierung ist es, die äußeren Kameraparameter, also die Lage und Orientierung der Kamera im Objektkoordinatensystem zu bestimmen. Da in dieser Arbeit der *Outside-in* Ansatz verfolgt wird, sind deshalb die Kameras zum Objekt und nicht zum Roboterkoordinatensystem zu kalibrieren. Für die 3D-Kalibrierung müssen die Brennweite f, Hauptpunktverschiebungen (c_x, c_y) und die Linsenverzerrungen (k_1, k_2) bereits bekannt sein (siehe Abschnitt 6.3.1).
Die Lage der Kamera relativ zum Endeffektor kann hier beliebig sein, allerdings muss sich der Endeffektor (bzw. seine Dioden) im Beobachtungsraum der Kameras befinden. Die einzige weitere Beschränkungen ist, dass das Objekt mit genauer Position und Orientierung ${}^{B}T_{O}$ im Arbeitsraum platziert ist, um eine hohe Genauigkeit zu erreichen (s. Abb. 6.14).
Unter Verwendung der äußeren Kameraparameter kann die Transformation eines Bildpunktes in das Roboterkoordinatensystem (Weltkoordinatensystem) durchgeführt werden, wodurch mit zwei Kameras die räumliche Vermessung von Leuchtpunkten am Endeffektor möglich wird. Dazu muss man die Lage der Kameras bezüglich des Objektes, also die Transformation des Kamerasystems in das Objektkoordinatensystem ${}^{O}T_{K}$ ermitteln(s. Abb. 6.14). Dazu sind eine Rotationsmatrix mit 9 Parametern und ein Positionsvektor mit 3 Parametern zu bestimmen. Diese Transformation bleibt konstant, wenn sich die Lage der Kameras relativ

zum Objekt nicht ändert, dieses wird jeweils durch eine starre Verbindung zwischen Objekt und Kamera gesichert.

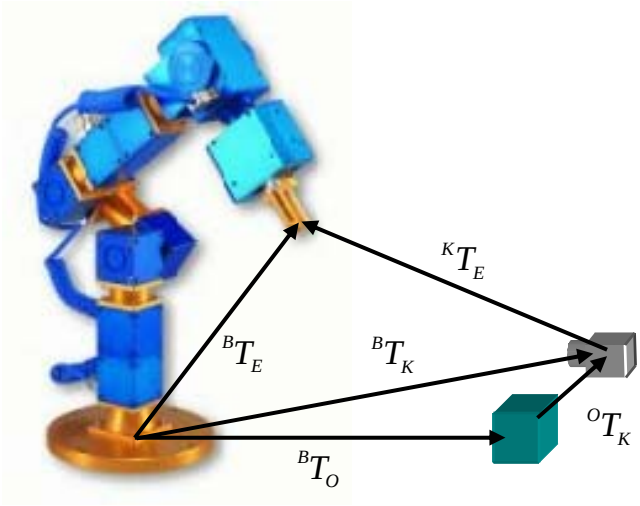


Abbildung 6.14: Die Transformationen

Die Lage der Kamera bzgl. des Objektes $^{O}T_{K}$ lässt sich durch folgende Gleichung bestimmen (s. Abb. 6.14):

$$\begin{aligned} {}^{B}T_{K} &= {}^{B}T_{O} \cdot {}^{O}T_{K} \\ \text{bzw.} \quad {}^{O}T_{K} &= ({}^{B}T_{O})^{-1} \cdot {}^{B}T_{K} \end{aligned} \tag{6.45}$$

Somit ist in dieser Gleichung gerade die Transformation vom Kamera- ins Weltkoordinatensystem $^{B}T_{K}$ unbekannt. Im Unterschied zur 2D-Kalibration darf die Transformation vom Welt- ins Kamerakoordinatensystem nun eine beliebig komplizierte Drehung um alle drei Achsen und eine Verschiebung (t_x, t_y, t_z) sein. Diese wird durch folgende Transformationsmatrix (s. Gl. 6.19 und Abb. 6.14) repräsentiert:

$$\begin{bmatrix} x_k \\ y_k \\ z_k \\ 1 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 & t_x \\ r_4 & r_5 & r_6 & t_y \\ r_7 & r_8 & r_9 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix} \quad \text{bzw. } {}^{K}P_{E} = {}^{K}T_{B} \cdot {}^{B}P_{E} \tag{6.46}$$

Die Elemente r_1 … r_9 der Transformationsmatrix sind, wie im Abschnitt 6.1 schon erwähnt wurde, die Elemente einer Rotationsmatrix und hängen daher nur von drei unabhängigen Parametern (α, β, γ) ab (s. Gl. 6.2).
Gleichsetzen von (6.19) und (6.21) ergibt den Zusammenhang zwischen Bild- und Weltkoordinaten:

$$\begin{aligned} x_f &= f \cdot \frac{r_1 x_w + r_2 y_w + r_3 z_w + t_x}{r_7 x_w + r_8 y_w + r_9 z_w + t_z} \\ y_f &= f \cdot \frac{r_4 x_w + r_5 y_w + r_6 z_w + t_y}{r_7 x_w + r_8 y_w + r_9 z_w + t_z} \end{aligned} \tag{6.47}$$

Von der anderen Seite werden die Sensorkoordinaten in die Bildkoordinaten mit der Gleichungen (6.22) und (6.23) überführt.

$$\begin{aligned} x_f &= (x_s - c_x) \cdot \left(1 + k \cdot (x_s - c_x)^2 + k \cdot (y_s - c_y)^2\right) \\ y_f &= (y_s - c_y) \cdot \left(1 + k \cdot (x_s - c_x)^2 + k \cdot (y_s - c_y)^2\right) \end{aligned} \tag{6.48}$$

Dabei werden die Hauptpunktverschiebungen und die Linsenverzerrungen der beiden Kameras berücksichtigt. Die Hauptpunktverschiebungen (c_x, c_y) und Verzerrungskoeffizient k wurden bereits im Abschnitt 6.3.1 ermittelt.
Die Bestimmung der äußeren Parameter verläuft in zwei Schritten:
Man schreibt die Gleichung (6.47) für x_f in folgender Form um:

$$x_f \cdot (r_7 x_w + r_8 y_w + r_9 z_w + t_z) = f \cdot (r_1 x_w + r_2 y_w + r_3 z_w + t_x) \tag{6.49}$$

Einer der acht unbekannten Parameter in der Gleichung (6.49) muss verschieden von Null und bekannt sein, da sonst die sich anschließende Optimierung lediglich die Nulllösung liefert. Die Kameras werden so an einer Strebe befestigt, dass sie die gleiche Position bzgl. der x-Achse haben, da die Strebe parallel zur X-Achse des Roboters steht. Bei der Ermittlung der inneren Parameter wurde bereits der Parameter t_x des Translationsvektors bestimmt und der wird für die Bestimmung der äußeren Kameraparameter weiterverwendet. Das System (6.49) kann also gelöst werden, wenn die Koordinaten von mindestens sieben Leuchtpunkten gemessen wurden, da sich sieben unbekannte Parameter in Gleichung (6.49) befinden. In unserem Fall wurden acht Leuchtpunkte, um die bessere Genauigkeit zu erreichen, gemessen. Diese Punkte haben konstante z-Werte und bilden eine achteckförmige Form der Endeffektortrajektorie im Weltkoordinatensystem (s. Abb. 6.15).

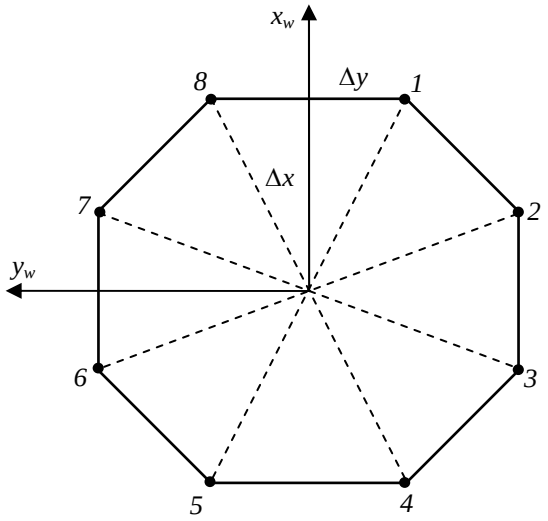


Abbildung 6.15: Die achteckförmige Trajektorie

Für acht Beobachtungspunkte kann man aus der Gleichung (6.49) ein System von acht Gleichungen bilden:

$$\begin{pmatrix} -x_w(1)f & -y_w(1)f & -z_w(1)f & x_f(1)x_w(1) & x_f(1)y_w(1) & x_f(1)z_w(1) & x_f(1) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -x_w(8)f & -y_w(8)f & -z_w(8)f & x_f(8)x_w(8) & x_f(8)y_w(8) & x_f(8)z_w(8) & x_f(8) \end{pmatrix} \cdot \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_7 \\ r_8 \\ r_9 \\ t_z \end{pmatrix} = \begin{pmatrix} t_x f \\ \vdots \\ t_x f \end{pmatrix} \quad (6.50)$$

Die Differenz zwischen den beiden Teilen des Systems der Gleichungen (6.50) wird mittels nichtlinearer Optimierung minimiert. Um die unerwünschte triviale Lösung auszuschließen, wurde eine zusätzliche Funktion mit 6 Nebenbedingungen für die Parameter der Rotationsmatrix, die nicht größer als 1 werden können (da $\cos(\alpha)\leq 1.0$ und $\sin(\alpha)\leq 1.0$), gestellt. Die Anfangswerte dieser Parameter wurden verschieden von Null festgelegt. Dabei wurde ein Teil der äußeren Parameter ermittelt. Der andere Teil der äußeren Parameter wird aus der Gleichung (6.47) für y_f bestimmt. Diese Gleichung kann man in folgender Form umschreiben:

$$y_f \cdot (r_7 x_w + r_8 y_w + r_9 z_w + t_z) = f \cdot (r_4 x_w + r_5 y_w + r_6 z_w + t_y) \quad (6.51)$$

Aus dieser Gleichung sind vier Parameter (r_4, r_5, r_6, t_y) zu ermitteln. Für die Lösung der Gleichung (6.51) werden die bereits zur Verfügung stehenden Bild- und Weltkoordinaten der 8 Leuchtpunkte benutzt. Das System, das zur Lösung benutzt wird, sieht demzufolge so aus:

$$\begin{pmatrix} x_w(1)f & y_w(1)f & z_w(1)f & f \\ \vdots & \vdots & \vdots & \vdots \\ x_w(8)f & y_w(8)f & z_w(8)f & f \end{pmatrix} \cdot \begin{pmatrix} r_4 \\ r_5 \\ r_6 \\ t_y \end{pmatrix} = \begin{pmatrix} y_f(1) \cdot (r_7 x_w(1) + r_8 y_w(1) + r_9 z_w(1) + t_z) \\ \vdots \\ y_f(8) \cdot (r_7 x_w(8) + r_8 y_w(8) + r_9 z_w(8) + t_z) \end{pmatrix} \quad (6.52)$$

Mittels nichtlinearer Optimierung der Gleichung (6.52) wurden die restlichen vier Parameter ermittelt. Dabei wurde eine zusätzliche Funktion mit 3 Nebenbedingungen für 3 Elemente der Rotationsmatrix in die Gl. (6.52) eingeführt. Mit diesem letzten Schritt sind alle äußeren Kameraparameter bekannt. Diese beschreiben die Position und Orientierung der Kamera im Weltkoordinatensystem ${}^{B}T_{K}$ vollständig. Gemäß der Gleichung (6.45) ist lediglich noch die inverse Transformation vom Objekt- ins Weltkoordinatensystem zu finden.

$${}^{B}T_{O} = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \Rightarrow \left({}^{B}T_{O}\right)^{-1} = \begin{pmatrix} 1 & 0 & 0 & -t_x \\ 0 & 1 & 0 & -t_y \\ 0 & 0 & 1 & -t_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (6.53)$$

Da die beiden Transformationsmatrizen in der Gleichung (6.45) jetzt bekannt sind, wird die Transformationsmatrix ${}^{O}T_{K}$, die die Position und die Orientierung der Kamera bzgl. des Objektes beschreibt, bestimmt. Für die Bestimmung der äußeren Kameraparameter nach dem ebenbeschriebenen Verfahren wurde ein C-Programm „ okp.c “ geschrieben und erfolgreich durch Experimente getestet. Die Ergebnisse der Ermittlung der äußeren Kameraparameter für beide Kameras sind in der Tabelle 6.2 zusammengefasst.

Parameter	Kamera 1	Kamera 2
Verschiebung in x - Richtung t_x	0.0	0.0
Verschiebung in y - Richtung t_y	-68.31	71.23
Verschiebung in z - Richtung t_z	38.4	37.32
Orientierung α	36.33	-33.32
Orientierung β	5.49	-0.24
Orientierung γ	-179.1	179.8

Tabelle 6.2: Die experimentell bestimmten äußeren Kameraparameter

Die berechneten äußeren Kameraparameter stimmen mit den durch Entfernungs- und Winkelmessung bei der Montage der Kameras ermittelten Parametern annährend überein. Die auf diese Weise bestimmten Kameraparameter werden auf die Genauigkeit der Bestimmung von Position und Orientierung des Objektes durch visuelle Information Auswirkungen haben. Dies kann quantitativ beim Stillstand des Objektes abgeschätzt werden. Entsprechendes wird im Kapitel 7 vorgenommen.

6.4 Bestimmung der Objektlage mittels visueller Information

Im Online-Betrieb der optisch geführten Robotersteuerung muss die Lage des Endeffektors bzgl. des Objektes ${}^{O}T_{E}$ (siehe Abb. 6.16) ständig aufgrund der gemessenen Signale der beiden Kameras über 4 nacheinander ein- und ausschaltender Laserdioden mittels eines Optimierungsverfahrens in Echtzeit bestimmt werden.

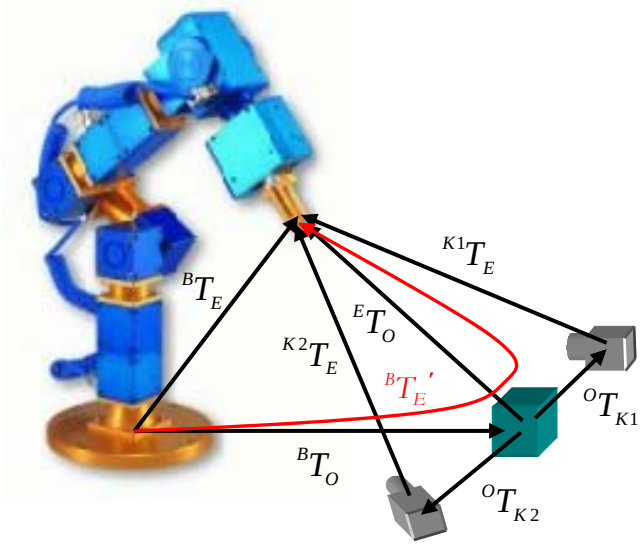


Abbildung 6.16: Die Transformationen zwischen den Objekt- , Kameras- , Endeffektor- und Basiskoordinatensystemen

Die Lage des Objektes bezüglich des Weltkoordinatensystems ${}^{B}T_{O}$ wurde in seinem Stillstand bereits im Primärvorgang mittels zwei am Rand des Arbeitsraumes festpositionierten Kameras grob ermittelt (siehe Abb. 2.2). Wenn die Transformationen ${}^{O}T_{E}$ und ${}^{B}T_{O}$ bekannt sind, wird die Position und Orientierung des Endeffektors ${}^{B}T_{E}^{'}$ ständig durch die Multiplikation dieser Transformationen berechnet. Wenn die Lage des Objektes ${}^{B}T_{O}$ sich ändert, wird sich dementsprechend auch die aus der visuellen Information berechnete Lage des Endeffektors bzgl. des Objektkoordinatensystems ${}^{O}T_{E}$ ändern. Dabei wird die Änderung der Lage des Objektes bei seiner Bewegung durch Änderung der Lage des Endeffektors im Weltkoordinatensystem erfasst, da die Lage des Objektes als unveränderliche Größe eingenommen wurde. Es werden also nach Abb. 2.3 die Veränderungen der Endeffektorlage im Weltkoordinatensystem $\vec{P}_{e}^{*}$ in Echtzeit berechnet. Da die Position und die Orientierung des Endeffektors durch 4 Dioden bestimmt werden müssen, wird zuerst die Position der ersten Laserdiode im Basiskoordinatensystem ${}^{B}P_{l1}$ ermittelt. Aus der Perspektive der ersten Kamera kann diese Position folgendermaßen bestimmt werden:

$$ {}^{B}P_{l1} = {}^{B}T_{K1} \cdot {}^{K1}P_{l1} \tag{6.54} $$

Aus der Perspektive der zweiten Kamera gilt bei gleicher Position:

$$ {}^{B}P_{l1} = {}^{B}T_{K2} \cdot {}^{K2}P_{l1} \tag{6.55} $$

Gleichsetzen von (6.54) und (6.55) liefert:

$$ {}^{B}T_{K1} \cdot {}^{K1}P_{l1} = {}^{B}T_{K2} \cdot {}^{K2}P_{l1} \tag{6.56} $$

bzw.:

$$ \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_{x1} \\ r_{14} & r_{15} & r_{16} & t_{y1} \\ r_{17} & r_{18} & r_{19} & t_{z1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^{k1}x_{l1} \\ {}^{k1}y_{l1} \\ {}^{k1}z_{l1} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{21} & r_{22} & r_{23} & t_{x2} \\ r_{24} & r_{25} & r_{26} & t_{y2} \\ r_{27} & r_{28} & r_{29} & t_{z2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^{k2}x_{l1} \\ {}^{k2}y_{l1} \\ {}^{k2}z_{l1} \\ 1 \end{bmatrix} \tag{6.57} $$

Die Lagen der einzelnen Kameras im Basiskoordinatensystem in der Gleichungen (6.56) und (6.57) kann man wie folgt berechnen:

$$ \begin{aligned} {}^{B}T_{K1} &= {}^{B}T_{O} \cdot {}^{O}T_{K1}; \\ {}^{B}T_{K2} &= {}^{B}T_{O} \cdot {}^{O}T_{K2}; \end{aligned} \tag{6.58} $$

Da die Transformationen von Kamera- in Objektkoordinatensystem aus den äußeren Kameraparameter bekannt und unveränderlich sind, und darüberhinaus die Lage des Objektes im Basiskoordinatesystem von Anfang an als bekannt vorausgesetzt wird, sind die Lagen der einzelnen Kameras im Basiskoordinatensystem gemäß der Gleichungen (6.58) ebenfalls bekannt.

Jetzt sind die Positionen der ersten Laserdiode bzgl. der einzelnen Kameras ${}^{K1}P_{l1}$ und ${}^{K2}P_{l1}$ zu bestimmen. Diese Positionen kann man aus der visuellen Information beider Kameras ermitteln. Die Transformation vom Kamera- ins Bildkoordinatensystem lässt sich aus der Gleichung (6.21) bestimmen:

$$x_{s1} = f_1 \cdot \frac{{}^{k1}x_{l1}}{{}^{k1}z_{l1}}; \qquad y_{s1} = f_1 \cdot \frac{{}^{k1}y_{l1}}{{}^{k1}z_{l1}};$$
$$x_{s2} = f_2 \cdot \frac{{}^{k2}x_{l1}}{{}^{k2}z_{l1}}; \qquad y_{s2} = f_2 \cdot \frac{{}^{k2}y_{l1}}{{}^{k2}z_{l1}}; \tag{6.59}$$

Die Gleichungen (6.59) lassen sich in folgender Form überführen:

$${}^{k1}x_{l1} = \frac{x_{s1} \cdot {}^{k1}z_{l1}}{f_1}; \qquad {}^{k1}y_{l1} = \frac{y_{s1} \cdot {}^{k1}z_{l1}}{f_1};$$
$${}^{k2}x_{l1} = \frac{x_{s2} \cdot {}^{k2}z_{l1}}{f_2}; \qquad {}^{k2}y_{l1} = \frac{y_{s2} \cdot {}^{k2}z_{l1}}{f_2}; \tag{6.60}$$

Setzt man die Gleichungen (6.60) in der Gleichung (6.57), so erhält man:

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_{x1} \\ r_{14} & r_{15} & r_{16} & t_{y1} \\ r_{17} & r_{18} & r_{19} & t_{z1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{x_{s1} \cdot {}^{k1}z_{l1}}{f_1} \\ \frac{y_{s1} \cdot {}^{k1}z_{l1}}{f_1} \\ {}^{k1}z_{l1} \\ 1 \end{bmatrix} = \begin{bmatrix} r_{21} & r_{22} & r_{23} & t_{x2} \\ r_{24} & r_{25} & r_{26} & t_{y2} \\ r_{27} & r_{28} & r_{29} & t_{z2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{x_{s2} \cdot {}^{k2}z_{l1}}{f_2} \\ \frac{y_{s2} \cdot {}^{k2}z_{l1}}{f_2} \\ {}^{k2}z_{l1} \\ 1 \end{bmatrix} \tag{6.61}$$

Die gemessenen Werte der Kamerasignale in der x- und y- Richtung x_{s1}, y_{s1}, x_{s2}, y_{s2} werden durch die Gl. (6.22-6.23) unter Berücksichtigung der Linsenverzerrung und der Bildhauptpunktveschiebung in der Form x_{f1}, y_{f1}, x_{f2}, y_{f2} berechnet. Nach der Multiplikation der Matrizen mit den Vektoren in der Gleichung (6.61) erhält man:

$${}^{k1}z_{l1} \cdot \left(r_{11} \cdot \frac{x_{f1}}{f_1} + r_{12} \cdot \frac{y_{f1}}{f_1} + r_{13} \right) + t_{x1} = {}^{k2}z_{l1} \cdot \left(r_{21} \cdot \frac{x_{f2}}{f_2} + r_{22} \cdot \frac{y_{f2}}{f_2} + r_{23} \right) + t_{x2};$$
$${}^{k1}z_{l1} \cdot \left(r_{14} \cdot \frac{x_{f1}}{f_1} + r_{15} \cdot \frac{y_{f1}}{f_1} + r_{16} \right) + t_{y1} = {}^{k2}z_{l1} \cdot \left(r_{24} \cdot \frac{x_{f2}}{f_2} + r_{25} \cdot \frac{y_{f2}}{f_2} + r_{26} \right) + t_{y2}; \tag{6.62}$$
$${}^{k1}z_{l1} \cdot \left(r_{17} \cdot \frac{x_{f1}}{f_1} + r_{18} \cdot \frac{y_{f1}}{f_1} + r_{19} \right) + t_{z1} = {}^{k2}z_{l1} \cdot \left(r_{27} \cdot \frac{x_{f2}}{f_2} + r_{28} \cdot \frac{y_{f2}}{f_2} + r_{29} \right) + t_{z2};$$

Daraus kann nun ein Gleichungssystem aufgestellt und gelöst werden:

$$\begin{pmatrix} r_{11} \cdot \frac{x_{f1}}{f_1} + r_{12} \cdot \frac{y_{f1}}{f_1} + r_{13} & -r_{21} \cdot \frac{x_{f2}}{f_2} - r_{22} \cdot \frac{y_{f2}}{f_2} - r_{23} \\ r_{14} \cdot \frac{x_{f1}}{f_1} + r_{15} \cdot \frac{y_{f1}}{f_1} + r_{16} & -r_{24} \cdot \frac{x_{f2}}{f_2} - r_{25} \cdot \frac{y_{f2}}{f_2} - r_{26} \\ r_{17} \cdot \frac{x_{f1}}{f_1} + r_{18} \cdot \frac{y_{f1}}{f_1} + r_{19} & -r_{27} \cdot \frac{x_{f2}}{f_2} - r_{28} \cdot \frac{y_{f2}}{f_2} - r_{29} \end{pmatrix} \cdot \begin{pmatrix} {}^{k1}z_{l1} \\ {}^{k2}z_{l1} \end{pmatrix} = \begin{pmatrix} t_{x2} - t_{x1} \\ t_{y2} - t_{y1} \\ t_{z2} - t_{z1} \end{pmatrix} \tag{6.63}$$

Nun ist die Position der Diode bzgl. der Kamera 1 (${}^{k1}z_{l1}$) und der Kamera 2 (${}^{k2}z_{l1}$) für die z-Richtung bekannt. Setzt man diese Werte in die Gleichungen (6.60) ein, so können die restlichen Werte der Position der Diode im Kamerakoordinatensystem für x- und y- Richtung

bestimmt werden. Für die anderen 3 Dioden können analoge Gleichungen aufgestellt und gelöst werden, um deren Positionen bzgl. der Kamera 1 und 2 zu bestimmen.
Am Beispiel der Diode 1 lässt sich seine Position im Objektkoordinatensystem bzgl. der Kamera 1 und 2 wie folgt bestimmen:

$$\begin{aligned} {}^{O}P_{l1}(1) &= {}^{O}T_{K1} \cdot {}^{K1}P_{l1} \\ {}^{O}P_{l1}(2) &= {}^{O}T_{K2} \cdot {}^{K2}P_{l1} \end{aligned} \tag{6.64}$$

Dabei unterscheiden sich die Werte beider Positionen nur geringfügig. Das ist damit verbunden, dass die Diode stets im besseren Sichtbereich einer der beiden Kameras liegt. Da sich im Online- Betrieb die Sichtbereich der Kameras nicht abschätzen lässt, um daraus die bestermittelte Position auszuwählen, wird die Position der Diode bzgl. des Objektes als mittlerer Wert bestimmt:

$${}^{O}P_{l1} = \frac{{}^{O}P_{l1}(1) + {}^{O}P_{l1}(2)}{2} \tag{6.65}$$

Dasselbe gilt auch für die anderen drei Dioden, um deren Positionen bzgl. des Objektes zu bestimmen.
Wie schon erwähnt wurde, schalten die Dioden nacheinander ein und wieder aus. Nach der Ausschaltung der 4. Diode wird eine Pause generiert. In dieser Pause ist die Position und Orientierung des Endeffektors im Basiskoordinatensystem mittels visueller Information der 4 Dioden zu ermitteln. Dazu werden die in der Abbildung 6.17 gezeigten Transformationen betrachtet. Zunächst untersucht man folgendes Dreieck: Objekt, Endeffektor, Diode 1.

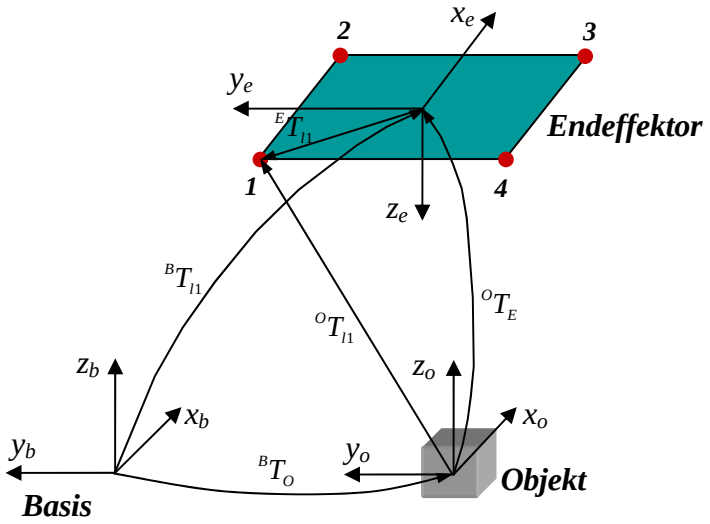


Abbildung 6.17: Die Transformationen zwischen den Objekt- , Endeffektor- (Dioden-) und Basiskoordinatensystemen

Die Position der Diode 1 bzgl. des Objektes lässt sich wie folgt bestimmen:

$${}^{O}P_{l1} = {}^{O}T_{E} \cdot {}^{E}P_{l1} \tag{6.66}$$

bzw.

$$\begin{bmatrix} {}^{o}x_{l1} \\ {}^{o}y_{l1} \\ {}^{o}z_{l1} \\ 1 \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 & t_x \\ r_4 & r_5 & r_6 & t_y \\ r_7 & r_8 & r_9 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} {}^{e}x_{l1} \\ {}^{e}y_{l1} \\ {}^{e}z_{l1} \\ 1 \end{bmatrix} \tag{6.67}$$

Daraus ergibt sich ein folgendes Gleichungssystem:

$$\begin{cases} {}^{o}x_{l1} = r_1 \cdot {}^{e}x_{l1} + r_2 \cdot {}^{e}y_{l1} + r_3 \cdot {}^{e}z_{l1} + t_x; \\ {}^{o}y_{l1} = r_4 \cdot {}^{e}x_{l1} + r_5 \cdot {}^{e}y_{l1} + r_6 \cdot {}^{e}z_{l1} + t_y; \\ {}^{o}z_{l1} = r_7 \cdot {}^{e}x_{l1} + r_8 \cdot {}^{e}y_{l1} + r_9 \cdot {}^{e}z_{l1} + t_z; \end{cases} \tag{6.68}$$

Entsprechende Gleichungssysteme können auch für die restlichen 3 Dioden aufgeschrieben werden. Es gilt also:

$$\begin{cases} {}^{o}x_{l2} = r_1 \cdot {}^{e}x_{l2} + r_2 \cdot {}^{e}y_{l2} + r_3 \cdot {}^{e}z_{l2} + t_x; \\ {}^{o}y_{l2} = r_4 \cdot {}^{e}x_{l2} + r_5 \cdot {}^{e}y_{l2} + r_6 \cdot {}^{e}z_{l2} + t_y; \\ {}^{o}z_{l2} = r_7 \cdot {}^{e}x_{l2} + r_8 \cdot {}^{e}y_{l2} + r_9 \cdot {}^{e}z_{l2} + t_z; \end{cases}$$

$$\begin{cases} {}^{o}x_{l3} = r_1 \cdot {}^{e}x_{l3} + r_2 \cdot {}^{e}y_{l3} + r_3 \cdot {}^{e}z_{l3} + t_x; \\ {}^{o}y_{l3} = r_4 \cdot {}^{e}x_{l3} + r_5 \cdot {}^{e}y_{l3} + r_6 \cdot {}^{e}z_{l3} + t_y; \\ {}^{o}z_{l3} = r_7 \cdot {}^{e}x_{l3} + r_8 \cdot {}^{e}y_{l3} + r_9 \cdot {}^{e}z_{l3} + t_z; \end{cases} \tag{6.69}$$

$$\begin{cases} {}^{o}x_{l4} = r_1 \cdot {}^{e}x_{l4} + r_2 \cdot {}^{e}y_{l4} + r_3 \cdot {}^{e}z_{l4} + t_x; \\ {}^{o}y_{l4} = r_4 \cdot {}^{e}x_{l4} + r_5 \cdot {}^{e}y_{l4} + r_6 \cdot {}^{e}z_{l4} + t_y; \\ {}^{o}z_{l4} = r_7 \cdot {}^{e}x_{l4} + r_8 \cdot {}^{e}y_{l4} + r_9 \cdot {}^{e}z_{l4} + t_z; \end{cases}$$

Die ersten Gleichungen (für *x*) in der Gleichungssystemen (6.68) und (6.69) können in ein anderes Gleichungssystem überführt werden:

$$\begin{cases} {}^{o}x_{l1} = r_1 \cdot {}^{e}x_{l1} + r_2 \cdot {}^{e}y_{l1} + r_3 \cdot {}^{e}z_{l1} + t_x; \\ {}^{o}x_{l2} = r_1 \cdot {}^{e}x_{l2} + r_2 \cdot {}^{e}y_{l2} + r_3 \cdot {}^{e}z_{l2} + t_x; \\ {}^{o}x_{l3} = r_1 \cdot {}^{e}x_{l3} + r_2 \cdot {}^{e}y_{l3} + r_3 \cdot {}^{e}z_{l3} + t_x; \\ {}^{o}x_{l4} = r_1 \cdot {}^{e}x_{l4} + r_2 \cdot {}^{e}y_{l4} + r_3 \cdot {}^{e}z_{l4} + t_x; \end{cases} \tag{6.70}$$

Für *y* gilt:

$$\begin{cases} {}^{o}y_{l1} = r_4 \cdot {}^{e}x_{l1} + r_5 \cdot {}^{e}y_{l1} + r_6 \cdot {}^{e}z_{l1} + t_y; \\ {}^{o}y_{l2} = r_4 \cdot {}^{e}x_{l2} + r_5 \cdot {}^{e}y_{l2} + r_6 \cdot {}^{e}z_{l2} + t_y; \\ {}^{o}y_{l3} = r_4 \cdot {}^{e}x_{l3} + r_5 \cdot {}^{e}y_{l3} + r_6 \cdot {}^{e}z_{l3} + t_y; \\ {}^{o}y_{l4} = r_4 \cdot {}^{e}x_{l4} + r_5 \cdot {}^{e}y_{l4} + r_6 \cdot {}^{e}z_{l4} + t_y; \end{cases} \tag{6.71}$$

Für *z* ergibt sich:

$$\begin{cases} {}^{o}z_{l1} = r_7 \cdot {}^{e}x_{l1} + r_8 \cdot {}^{e}y_{l1} + r_9 \cdot {}^{e}z_{l1} + t_z; \\ {}^{o}z_{l2} = r_7 \cdot {}^{e}x_{l2} + r_8 \cdot {}^{e}y_{l2} + r_9 \cdot {}^{e}z_{l2} + t_z; \\ {}^{o}z_{l3} = r_7 \cdot {}^{e}x_{l3} + r_8 \cdot {}^{e}y_{l3} + r_9 \cdot {}^{e}z_{l3} + t_z; \\ {}^{o}z_{l4} = r_7 \cdot {}^{e}x_{l4} + r_8 \cdot {}^{e}y_{l4} + r_9 \cdot {}^{e}z_{l4} + t_z; \end{cases} \tag{6.72}$$

Die letzen drei Gleichungssysteme können in Matrizenform gebildet werden:

$$\begin{bmatrix} {}^{o}x_{l1} \\ {}^{o}x_{l2} \\ {}^{o}x_{l3} \\ {}^{o}x_{l4} \end{bmatrix} = \begin{bmatrix} {}^{e}x_{l1} & {}^{e}y_{l1} & {}^{e}z_{l1} & 1 \\ {}^{e}x_{l2} & {}^{e}y_{l2} & {}^{e}z_{l2} & 1 \\ {}^{e}x_{l3} & {}^{e}y_{l3} & {}^{e}z_{l3} & 1 \\ {}^{e}x_{l4} & {}^{e}y_{l4} & {}^{e}z_{l4} & 1 \end{bmatrix} \cdot \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ t_x \end{bmatrix};$$

$$\begin{bmatrix} {}^{o}y_{l1} \\ {}^{o}y_{l2} \\ {}^{o}y_{l3} \\ {}^{o}y_{l4} \end{bmatrix} = \begin{bmatrix} {}^{e}x_{l1} & {}^{e}y_{l1} & {}^{e}z_{l1} & 1 \\ {}^{e}x_{l2} & {}^{e}y_{l2} & {}^{e}z_{l2} & 1 \\ {}^{e}x_{l3} & {}^{e}y_{l3} & {}^{e}z_{l3} & 1 \\ {}^{e}x_{l4} & {}^{e}y_{l4} & {}^{e}z_{l4} & 1 \end{bmatrix} \cdot \begin{bmatrix} r_4 \\ r_5 \\ r_6 \\ t_y \end{bmatrix}; \tag{6.73}$$

$$\begin{bmatrix} {}^{o}z_{l1} \\ {}^{o}z_{l2} \\ {}^{o}z_{l3} \\ {}^{o}z_{l4} \end{bmatrix} = \begin{bmatrix} {}^{e}x_{l1} & {}^{e}y_{l1} & {}^{e}z_{l1} & 1 \\ {}^{e}x_{l2} & {}^{e}y_{l2} & {}^{e}z_{l2} & 1 \\ {}^{e}x_{l3} & {}^{e}y_{l3} & {}^{e}z_{l3} & 1 \\ {}^{e}x_{l4} & {}^{e}y_{l4} & {}^{e}z_{l4} & 1 \end{bmatrix} \cdot \begin{bmatrix} r_7 \\ r_8 \\ r_9 \\ t_z \end{bmatrix};$$

In diesen Gleichungssystemen sind die Positionen der Dioden im Objektkoordinatensystem bekannt. Ebenso sind die Positionen der Dioden bzgl. des Endeffektors bekannt, da diese auf dem Endeffektor mit bekanntem Abstand zum Endeffektorkoordinatensystem platziert sind.
Damit können im Gleichungssystem (6.73) die Vektoren $[r_1\ r_2\ r_3\ t_x]$, $[r_4\ r_5\ r_6\ t_y]$ und $[r_7\ r_8\ r_9\ t_z]$ bestimmt werden. Sie beschreiben Position und Orientierung des Endeffektors im Objektkoordinatensystem.
Nach der Lösung des Gleichungssystems (6.73) ist die Transformationsmatrix zur Beschreibung der Position und Orientierung des Endeffektors im Objektkoordinatensystem bekannt. Jetzt kann die Transformationsmatrix zur Beschreibung der Position und Orientierung mittels visueller Information ($^{B}T_{E}^{*}$) wie folgt berechnet werden:

$$^{B}T_{E}^{*} = {}^{B}T_{O} \cdot {}^{O}T_{E} \tag{6.74}$$

Wenn das Objekt sich bewegt, bildet sich eine Differenz zwischen der mittels der direkten Kinematik und der visuellen Information berechneten Position und Orientierung des Endeffektors (s. Abb. 2.3), die verschieden von 0 ist, aus:

$$^{B}T_{E} - {}^{B}T_{E}^{*} \neq 0 \tag{6.75}$$

Um diese Differenz abzubilden, müssen die Taktzeiten der direkten Kinematik und des Kameramodells gleich sein. Da das Kameramodell 10-mal langsamer (100ms) als sie offene Kette der Robotersteuerung (10ms) funktioniert, wird eine derartige Differenz mit einer Taktzeit von 100ms berechnet.

Aufgrund der Berechnungs- und Messfehler (Messstörungen) tritt stets eine Differenz auf, die wesentlich bei der Objektverfolgung vergrößert wird. Durch Bestimmung des Messfehlers im Stillstand des Objektes (siehe Kapitel 7), wird die Größe der Schranke (siehe Abb. 2.3) bestimmt. Unter Berücksichtigung der Schranke bildet die Differenz (nach Gl. 6.75) bei der Objektverfolgung eigentlich die Bewegung des Objektes im Arbeitsraum ab:

$$ {}^{B}T_{E} - {}^{B}T_{E}^{*} = \Delta {}^{B}T_{O} \tag{6.76} $$

Diese Differenz dient zur Generierung einer Trajektorie, um die Korrektur der Bewegung des Endeffektors bei der Verfolgung des Objektes (Tracking) vorzunehmen. Im Block *Trajektoriengenerierung 2* (s. Abb. 2.3) wird eine räumliche Trajektorie (s. Abb. 4.8) innerhalb eines Kameramodelltaktes (100ms) mit einer Abtastzeit von 10ms generiert. Diese Trajektorie dient zur Korrektur der Roboterbewegung für die Objektverfolgung.
Für die Bestimmung der Position und der Orientierung des Endeffektors im Bezug auf das Basiskoordinatensystem auf Grund visueller Information gemäß dem oben beschriebenen Verfahren wurde ein C-Programm „ visual.c “ geschrieben. Im nächsten Kapitel wird dieses Programm experimentell getestet und die verschiedenen Verifikationen der gesamten optisch geführten Robotersteuerung durchgeführt.

Kapitel 7

Ergebnisse experimenteller Untersuchungen der gesamten Struktur der optisch geführten Robotersteuerung

Da die direkte und inverse Kinematik des Roboters, der Trajektoriengenerator und das Kameramodell bereits ermittelt wurden, existieren alle Voraussetzungen, um die gesamte Verarbeitungskette der optisch geführten Robotersteuerung mit einer visuellen Rückführung zu schließen und experimentell zu untersuchen (siehe Abb. 2.3). Im Laufe dieser experimentellen Untersuchungen sind folgende Fragen zu klären:

- Die Abweichungen zwischen visuell ermittelter und realer Objektlage im Stillstand müssen ermittelt werden, um die Richtigkeit und Genauigkeit der ermittelten Kameraparameter sowie des Verfahrens zur Bestimmung der Objektlage zu überprüfen.
- Die Funktionalität der entwickelten optisch geführten Robotersteuerung bei Objektverfolgung mit verschiedenen Geschwindigkeiten ist zu untersuchen. Die maximal erreichbare Geschwindigkeit bei der Objektverfolgung hängt dabei sehr stark von der Abtastzeit des Kameramodells ab. Wie schon im Kapitel 6 beschrieben wurde, beträgt sie in unserem Fall 10 Hz.

Dabei müssen die Fragen der Stabilität und der Robustheit dieser Steuerung bei Vorgabe verschiedener Größen der Beschränkung und der maximalen Geschwindigkeit sowie der Beschleunigung des Roboters untersucht werden.
Wie schon im Kapitel 2 erwähnt wurde, wird in dieser Arbeit eine unmittelbare Rückführung in Form eines P- Reglers (von Anfang an $k=1$) für die optisch geführte Steuerung ausgewählt, da keine Möglichkeit besteht, die Analyse des stark nichtlinearen Systems (Roboterkinematik, -dynamik und Kameramodell) durchzuführen und einen passenden nichtlinearen Regler zu entwerfen. Allerdings könnte zwischen einfachen Reglern (P, PI, PID) variiert werden. Derartige Untersuchungen werden aber in dieser Arbeit nicht vorgenommen.
Nach der Implementierung der Software im Signalprozessorsystem (siehe dazu Anhang A) wird auf die oben beschriebenen Untersuchungspunkte eingegangen.

7.1 Stillstandsanalyse

Gemäß der Struktur der optisch geführten Robotersteuerung in der Abb. 2.3 wird zuerst für einen vorgegebenen relativen Abstand zwischen dem Objekt und dem Endeffektor $\Delta\vec{P}_{soll}$

mit Hilfe des Blocks *Trajektoriengenerierung 1* eine räumliche Trajektorie generiert, die den Endeffektor mit diesem gewünschten Abstand vom Objekt positioniert. Danach wird die optische Rückführung eingeschaltet, um die visuelle Beobachtung des Objektes vorzunehmen. Dabei werden die Größen der Beschränkung für das Differenzsignal $\Delta\vec{P}_o$ zwischen den realen und visuellen Werten der Lage des Endeffektors im Arbeitsraum zunächst auf $-2\text{mm} \leq \Delta\vec{P}_o \leq 2\text{mm}$ und $-2° \leq \Delta\vec{P}_o \leq 2°$ festgelegt. Dieses Differenzsignal bildet im idealen Fall die Bewegung des Objektes im Weltkoordinatensystem ab. Im Stillstand des

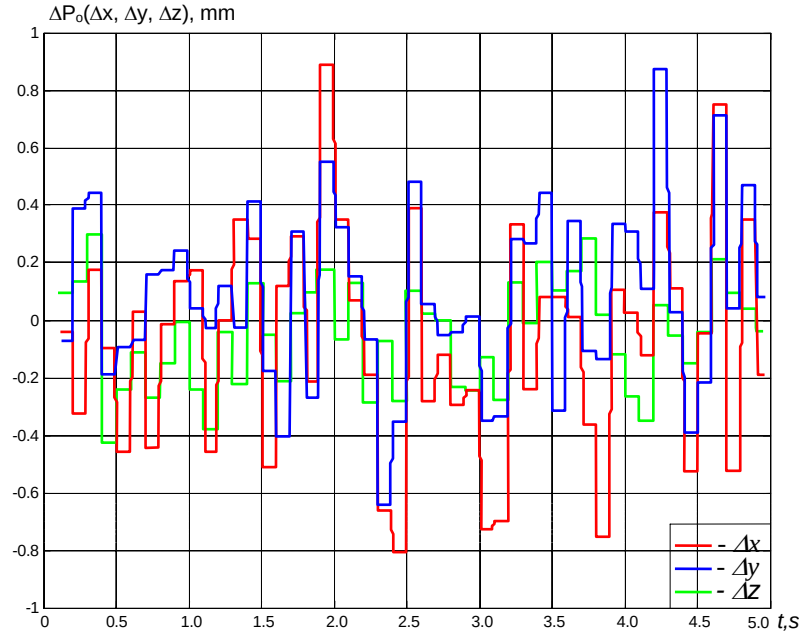


Abbildung 7.1: Die Abweichungen der realen von der visuell ermittelten Objektposition im Stillstand

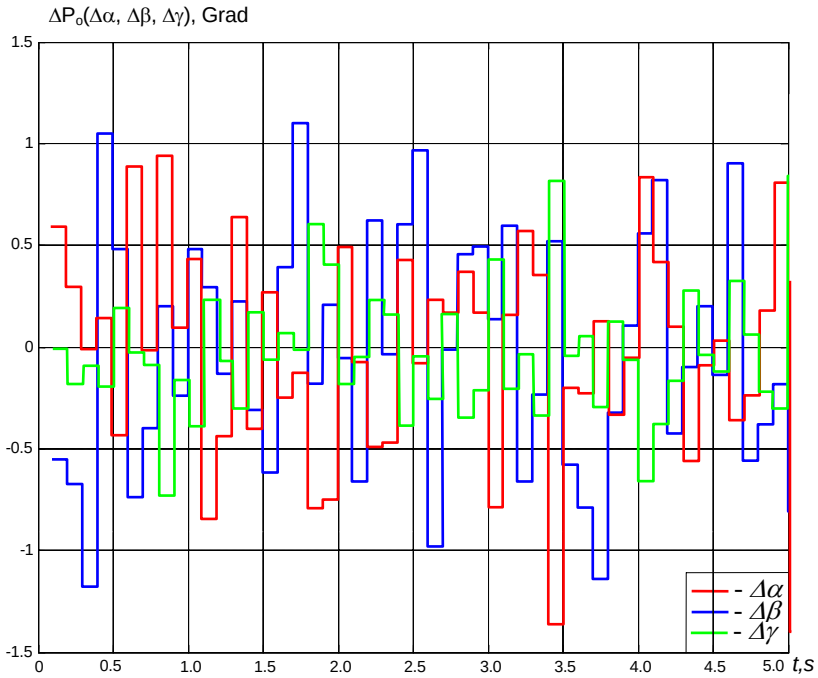


Abbildung 7.2: Die Abweichungen der realen von der visuell ermittelten Objektorientierung im Stillstand

Objektes, wenn seine Lage genau bekannt ist, müsste dieses Signal Null sein. In Realität treten jedoch Störungen auf, beispielsweise bei der Messung der visuellen Signale der Laserdioden wegen der ungleichmäßigen Beleuchtung des Arbeitsraums, bei der Reflexion des Lichtes der Laserdioden von der Roboteroberfläche und bzgl. der Wirkung des Restlichtes der Laserdioden bei deren Modulation. Sie verhindern ein Verschwinden der Differenzsignale. Im Stillstand des Objektes wurden diese Differenzsignale $\Delta\vec{P}_o$ bzgl. der Position (siehe Abb. 7.1) und der Orientierung (siehe Abb. 7.2) gemessen. Sie sind auch als Abweichungen der realen von der visuell ermittelten Objektlage interpretierbar. Wie man erkennt, sind die maximalen Abweichungen für die Position nicht größer als 1mm und für die Orientierung nicht größer als 1.5°. Sie liegen damit unter der vorerst festgelegten Schranke von ±2mm. Die Fehler für die Bestimmung der Lage des Endeffektors auf Grund visueller Information mittels des Kameramodells sind damit kleiner als diese Störungen. Damit ist eine gute Genauigkeit bei der Ermittlung der Kameraparameter und der Objektlage nachgewiesen. Da die Größe der Schranke größer als das maximale Differenzsignal $\Delta\vec{P}_o$ festgelegt wurde, bewegt sich der Roboter nicht.

Um die Stabilität und die Reaktion der optisch geführten Robotersteuerung auf visuelle Störungen hin zu prüfen, wird in einem weiteren Experiment die Größe der Schranke auf Null gesetzt. Dabei befindet sich das Objekt im Stillstand. Danach wurden die Abweichungen der realen von der visuell ermittelten Objektposition $\Delta\vec{P}_o$ ermittelt, das Ergebnis zeigt Abbildung 7.3. Aus dieser Abbildung sieht man, dass sich die Genauigkeit bei der Ermittlung der Position des Objektes im Vergleich zum Versuch mit Beschränkung verschlechterte. Das ist damit zu erklären, dass der Roboter auf die genannten Störungen reagiert und Schwingungen des Endeffektors in verschiedenen Richtungen verursacht. Wegen dieser Bewegungen des Endeffektors vergrößert sich letztendlich der Fehler bei der Bestimmung der Objektlage. Eine ähnliche Verschlechterung des Differenzsignals $\Delta\vec{P}_o$ entsteht auch bei der Ermittlung der Objektorientierung. Der maximale Fehler beträgt dabei 2.1°. Aus der Abbildung 7.3 ist ersichtlich, dass das System der optisch geführten Robotersteuerung mit stabilem Verhalten auf die Störungen unter 1mm bzgl. der Objektposition reagiert.

Um die Robustheit des Systems bei größeren Störungen zu überprüfen, wird einer der kinematischen (Denavit- Hartenberg) Parameter in der direkten Kinematik, z.B. d_5 in Tabelle 5.1, verändert. Dieser Parameter wurde 2mm kleiner (d_5=*303mm*) im Vergleich zum realen Wert (d_5=*305mm*) festgelegt. Dies führt zur Veränderung der Lage des Endeffektors bzgl. der realen Werte, also zu einer zusätzlichen statischen Störung durch Modellfehler im Regelkreis. Tabelle 7.1 zeigt die Verschiebung der Lage des Endeffektors bei den oben genannten Änderungen einer der kinematischen Parameter. Dabei wurden die Abweichungen zwischen der gestörten realen und der visuell ermittelten Objektposition $\Delta\vec{P}_o$ gemessen und in der Abbildung 7.4 dargestellt. Diese Abbildung zeigt die zu erwartende Verschlechterung der Positionssignale im Vergleich zur Abb. 7.3. Das gesamte System bleibt aber dennoch stabil. Danach wurde die Abweichung des Parameters d_5 soweit erhöht, bis das System instabil wurde. Dabei war es erforderlich den Parameter d_5 um 5mm kleiner (d_5=*300mm*) im Vergleich zum realen Wert festzulegen. In unserem Fall wurde exemplarisch die Veränderung des Parameters d_5 gewählt. Allerdings ist es prinzipiell unerheblich, welcher der kinematischen Parameter mit Fehlern behaftet wird, da letztendlich die Veränderung der

Lage des Endeffektors	X_e	Y_e	Z_e	α_e	β_e	γ_e
Ohne Veränderung des Parameters d_5=305mm	484.05	-29.37	256.93	179.44°	-0.32°	89.18°
Mit der Veränderung des Parameters d_5=303mm	482.36	-28.37	257.13	179.45°	-0.32°	89.184°

Tabelle 7.1: Die Veränderung der Lage des Endeffektors bei der Änderung der kinematischen Parameter

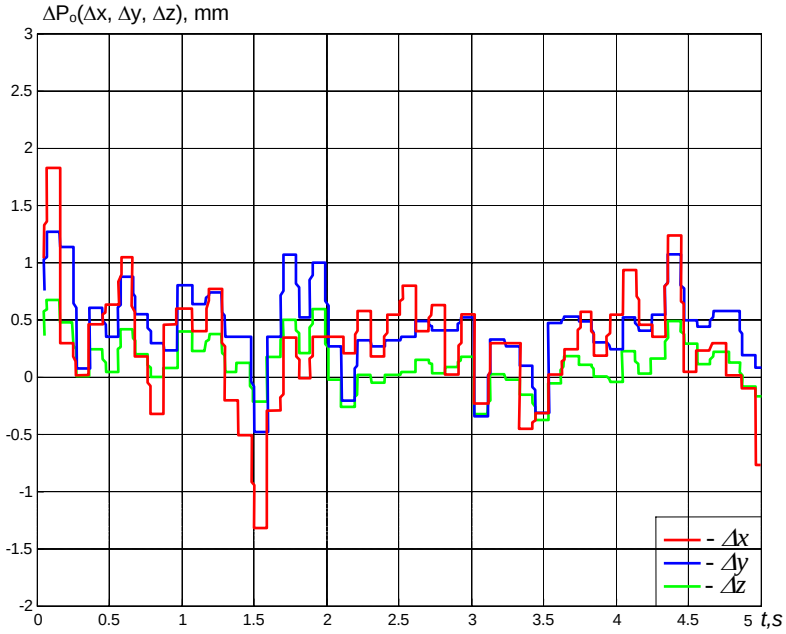


Abbildung 7.3: Die Abweichungen der realen von der visuell ermittelten Objektposition im Stillstand ohne Schranke

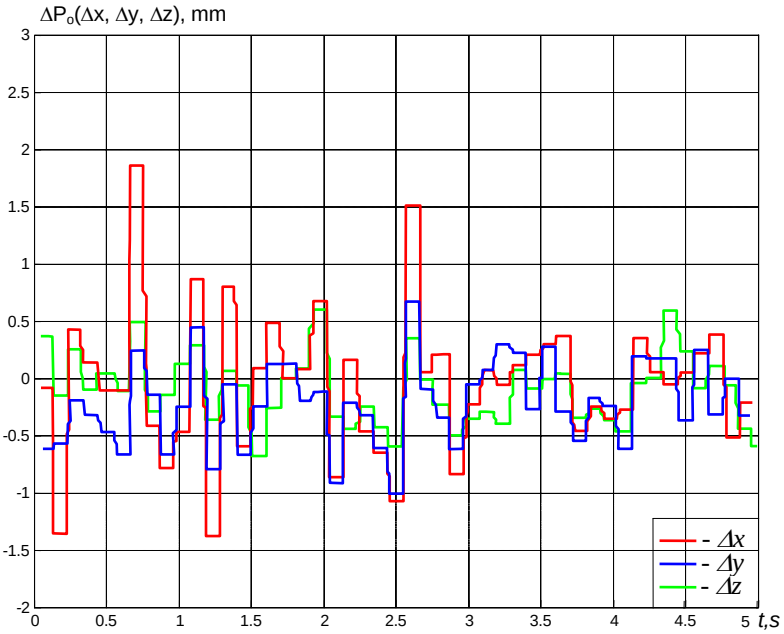


Abbildung 7.4: Die Abweichungen der gestörten realen von der visuell ermittelten Objektposition im Stillstand ohne Schranke

Roboterlage von Bedeutung ist. Wenn alle kinematischen Parameter mit kleinen Fehlern behaftet werden, wird dies ebenfalls zur Veränderung der Lage des Endeffektors führen. Allerdings zeigt sich, dass die Orientierung des Endeffektors viel stärker bei der Änderung einer der Parameter von den vorgegebenen Werten abweicht. Erst als alle kinematischen Parameter um 1mm verkleinert wurden, wurde das Gesamtsystem instabil. Dies weist eine ausreichende Robustheit des Systems zur optisch geführten Robotersteuerung im Stillstand des Objektes nach.
Um die Messstörungen der Kameras für die Robotersteuerung auszufiltern, muss die Größe der Schranke festgelegt werden. Im Stillstand des Objektes wurde sie bei 1mm für die Position und 1.5° für die Orientierung festgelegt. Damit kann der zweite Abschnitt der experimentellen Untersuchungen, nämlich die Objektverfolgung mit verschiedenen Geschwindigkeiten, betrachtet werden.

7.2 Bewegungsanalyse

Da für die verschiedenen Anwendungen unterschiedliche Geschwindigkeiten bei der Objektverfolgung erforderlich sind und die erzielbaren Ergebnisse stark von dieser Geschwindigkeit abhängen, müssen letztendlich die experimentellen Untersuchungen der gesamten optisch geführten Robotersteuerungen bei verschiedenen Objektgeschwindigkeiten durchgeführt werden. Dabei sind die Grenzen einer derartigen Steuerung geschwindigkeitsbezüglich mit der im Kapitel 3 beschriebenen Hardware aufzuzeigen. Diese Untersuchungen werden mit den verschiedenen Einstellungen (Größe der Schranke, Größe der maximalen Geschwindigkeit und der Beschleunigung des Roboters) durchgeführt, um die optimalen Parameter der Steuerung auszuwählen.
Da sich das Objekt sehr oft für industriellen Zwecke entlang einer kartesischen Trajektorie (die Position und Orientierung des Objektes wird damit geändert) bewegen soll, muss die entwickelte optisch geführte Robotersteuerung in der Lage sein, die Verfolgung einer derartigen Trajektorie bezüglich des Roboterendeffektors zu realisieren. Dazu wurde das Objekt zuerst mit langsamen Geschwindigkeiten handgeführt bewegt. Dabei wurde festgestellt, dass der Roboter solchen Trajektorien in verschiedenen Richtungen folgen kann. Bei Erhöhung der Objektgeschwindigkeit wurde das System jedoch instabil. Durch eine derartige Untersuchung des Objektes können keine qualitativen Aussagen über mögliche Geschwindigkeiten und über den bei der Objektverfolgung entstehenden Fehler gemacht werden.
Um das Objekt mit verschiedenen, definierten Geschwindigkeiten zu fahren, wurde ein entlang der linearen Achse gesteuerter Schlitten verwendet (siehe Abb. 3.6). Dieser wird durch eine speicherprogrammierbare Steuerung (SPS) entlang der *x*- Achse gesteuert. In Abbildung 7.5 sind die programmierten Strecken der Schlittenbewegungen dargestellt. In Tabelle 7.2 sind die Geschwindigkeiten des Schlittens für die *Experimente 1* und *2* zusammengefasst dargestellt. Für das *Experiment 1* wurden relativ langsame Geschwindigkeiten (unter 100mm/s) festgelegt, um die Objektverfolgung zuerst bei solchen Geschwindigkeiten zu testen. Um die Grenzen der entwickelten Steuerung geschwindigkeitsbezüglich aufzuzeigen, wurden für das *Experiment 2* größere Geschwindigkeiten (über 100mm/s) festgelegt.

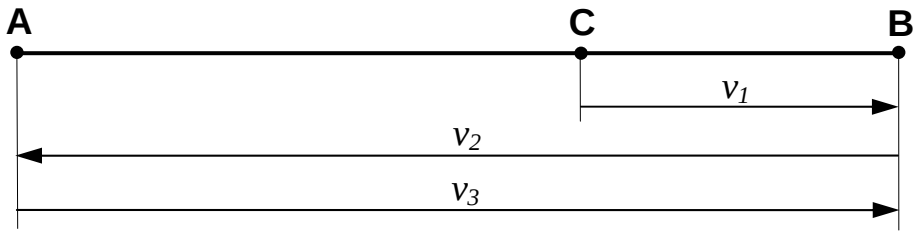


Abbildung 7.5: Die Strecken der Schlittenbewegungen mit verschiedenen Geschwindigkeiten v_i

Strecke	Experiment 1	Experiment 2
C→B	$v_1=50mm/s$	$v_1=100mm/s$
B→A	$v_2=75mm/s$	$v_2=150mm/s$
A→B	$v_3=100mm/s$	$v_3=200mm/s$

Tabelle 7.2: Die Geschwindigkeiten des Schlittens für die verschiedenen Strecken der Bewegungen

Durch die handgeführten Experimente wurde festgestellt, dass die genauer zu untersuchende Objektverfolgung prinzipiell in beliebiger Richtung erfolgen kann. Da sich der Schlitten entlang einer Achse (z.B. *x*- Achse) bewegen kann, werden die weiteren Experimente nur entlang einer Richtung durchgeführt. Für das *Experiment 1* sind die Bewegungsverläufe des am Schlitten befestigten Objektes entlang der *x*- Achse $x_{ogew.}$ und die mit Hilfe der visuellen Regelung ermittelten $P_o = (x_{oerm.}, y_{oerm.}, z_{oerm.})$ in der Abbildung 7.6 dargestellt.

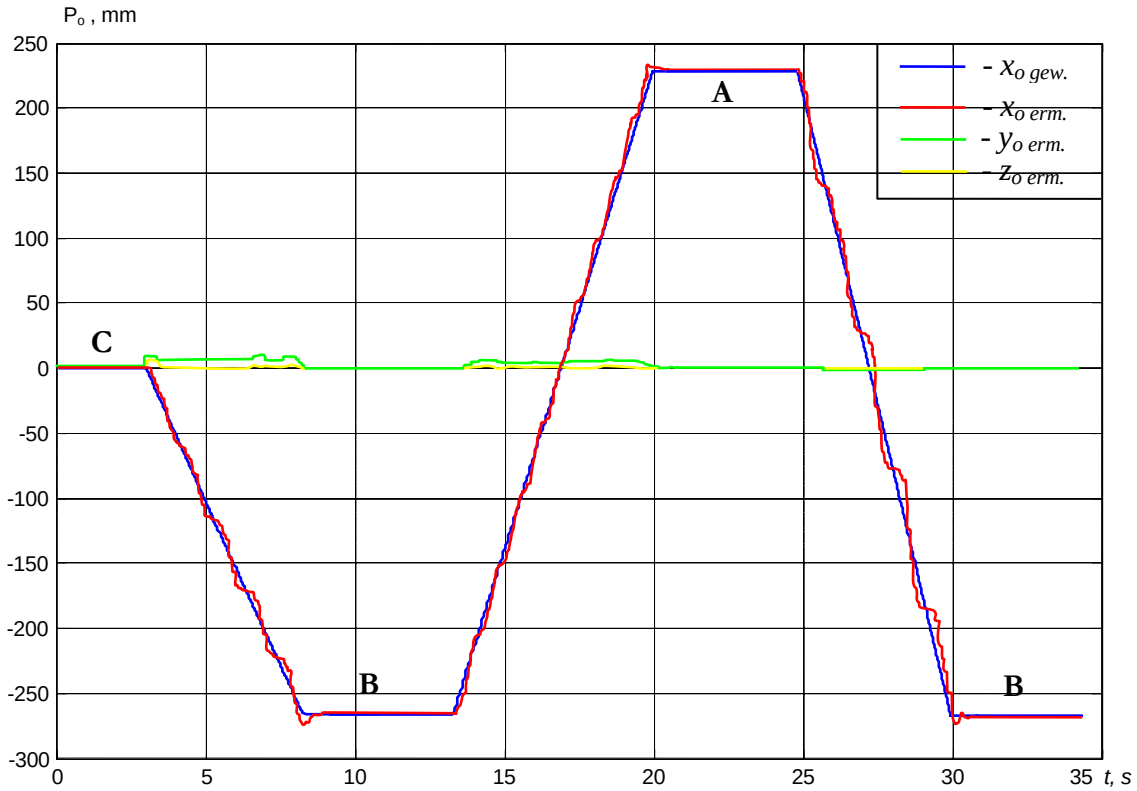


Abbildung 7.6: Der gewünschte und der visuell ermittelte Bewegungsverlauf des Objektes entlang der x- Achse mit den für den Experiment 1 vorgegebenen Geschwindigkeiten und mit einer Schranke ±1mm

Die Größe der Beschränkung wurde dabei mit 1mm für die Position und 1.5° für die Orientierung festgelegt. Die Geschwindigkeit und die Beschleunigung des Roboters wurden auf ein Viertel der maximal erlaubten Werte eingestellt. Damit werden dieselben Einstellungen gewählt, die bereits für den Stillstand des Objektes am besten geeignet waren. Die Abbildung 7.6 zeigt, dass sich die visuell ermittelten und die realen Bewegungen des Objektes mit den für das *Experiment 1* festgelegten Geschwindigkeiten auf der Strecke ***C→B*** bereits wesentlich unterscheiden. Das ist jedoch zu einem beträchtlichen Teil dadurch verursacht, dass sich der Schlitten mit dem Objekt bei diesen langsamen Geschwindigkeiten nicht ruckfrei bewegt. In der Abb. 7.6 sind dagegen die gewünschten (programmierten) Bewegungen des Objektes dargestellt. Deswegen ist es sehr schwierig bei langsamen Geschwindigkeiten die Genauigkeit der Verfolgung des Roboters abzuschätzen. Bei der Erhöhung der Geschwindigkeit des Schlittens wird die Bewegung des Objektes fast ruckfrei und damit der gewünschten sehr ähnlich. Dies ist z.B. auch aus dem Fehler zwischen den gewünschten und den visuell ermittelten Bewegungen des Objektes bzgl. der *y*- und *z*- Achse ersichtlich. Da das Objekt sich entlang der *x*- Achse bewegt, müssen die Positionen des Objektes entlang der *y*- und *z*- Achse im Idealfall gleich Null sein. Die Abb. 7.6 zeigt, dass bei der Erhöhung der Geschwindigkeit des Schlittens diese Fehler kleiner werden. So verringert sich auf der Strecke ***B→A*** der sichtbare Fehler im Vergleich zur Strecke ***C→B***. In Wirklichkeit jedoch vergrößert sich der Verfolgungsfehler des Roboters bei der Erhöhung der Geschwindigkeit. Diese Feststellung bekräftigt der visuell ermittelte Bewegungsverlauf des Objektes auf der Strecke ***A→B***. Der Fehler bzgl. der *x*- Achse wird auf dieser Strecke auf Grund der Geschwindigkeitserhöhung im Vergleich zur Strecke ***B→A*** sichtbar größer.

Es ist auch von Interesse, ob die gewählten Einstellungen bzgl. der Beschränkung sowie der Geschwindigkeit und der Beschleunigung des Roboters wirklich optimal sind. Dazu wird zuerst die Größe der Beschränkung bzgl. der Position verändert und mit $-0.5mm \le \Delta\vec{P}_o \le 0.5mm$ festgelegt. Die Abbildung 7.7 zeigt hierfür den Unterschied zwischen gewünschtem und visuell ermitteltem Bewegungsverlauf entlang der *x*- Achse mit den für den *Experiment 1* festgelegten Geschwindigkeiten. Die Analyse des Verfolgungsfehlers in Abbildung 7.7 bestätigt im Vergleich zu Messungen in Abb. 7.6, dass sich die Verfolgungsgenauigkeit aufgrund der Verkleinerung der Positionsbeschränkung ($-0.5mm \le \Delta\vec{P}_o \le 0.5mm$) verschlechtert. Im Stillstand des Objektes (**B**) sieht man die Schwankungen der Roboterposition. Das Gesamtsystem der optisch geführten Robotersteuerung erweist sich dabei sehr unruhig, bis es letztendlich auf der Strecke ***B→A*** instabil wird. Dies zeigt, dass die Größe der Beschränkung bei der Bewegung des Objektes nicht kleiner als die in seinem Stillstand optimal ermittelte sein darf.

Die Einstellungen der maximalen Geschwindigkeit und der Beschleunigung des Roboters haben ebenfalls einen großen Einfluss auf die Verfolgungsgenauigkeit auf. Von Anfang an wurden diese Werte auf ein Viertel der maximal erlaubten Werte festgelegt. Dabei ist es wichtig zu untersuchen, wie sich die Erhöhung dieser Werte z.B. bis zur Hälfte der maximal erlaubten Geschwindigkeit und Beschleunigung des Roboters auf die Verfolgungsgenauigkeit auswirkt. Die Abbildung 7.8 zeigt den gewünschten und den visuell ermittelten Bewegungsverlauf des Objektes mit den für *Experiment 1* festgelegten Geschwindigkeiten und mit den oben beschriebenen Einstellungen der Robotergeschwindigkeit und -beschleunigung. Entlang der Strecke ***C→B*** zeigt das System ausreichende Verfolgungsgenauigkeit. Im

Stillstand des Objektes wurde das System instabil und die Objektverfolgung auf der Strecke ***B→A*** ist nicht mehr gewährleistet.

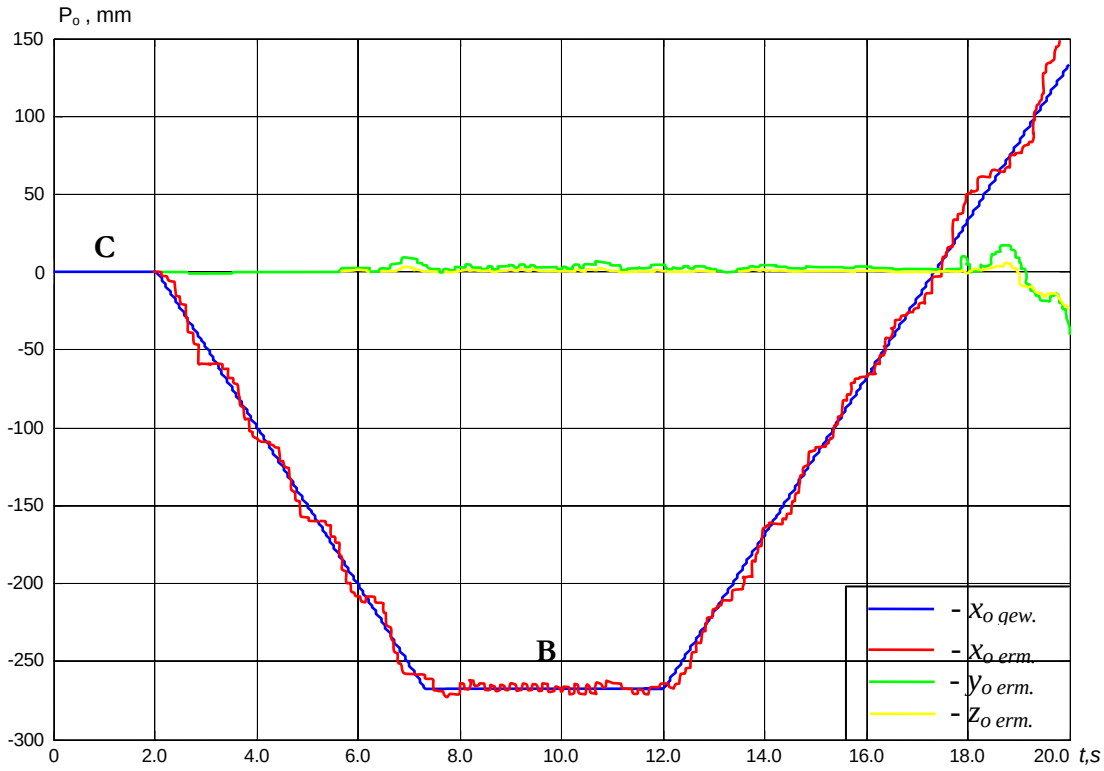


Abbildung 7.7: Der gewünschte und der visuell ermittelte Bewegungsverlauf des Objektes entlang der x- Achse mit den für den Experiment 1 vorgegebenen Geschwindigkeiten und mit einer Schranke ±0.5mm

Um zu untersuchen, bis zu welchen Geschwindigkeiten der Objektverfolgung des Systems der optisch geführten Robotersteuerung funktioniert, wurde das Objekt mit für den in *Experiment 2* festgelegten Geschwindigkeiten bewegt. Zuerst wurde das Objekt entlang der *x*-Achse auf den Strecken zwischen -270mm und 230mm bewegt. Dabei wurde das Gesamtsystem auf der Strecke ***B→A*** instabil und die Objektverfolgung konnte nicht mehr gewährleistet werden. Der gewünschte und der visuell ermittelte Bewegungsverlauf des Objektes ähnelt dabei der in der Abb. 7.7 dargestellten. Als Ursache dafür wird die Ansammlung des Fehlers vermutet, die auf weiteren Strecken zu den Instabilitäten des Gesamtsystems führt. Deswegen wurde die Strecke der Objektbewegungen verkürzt (-100mm $\leq P_o \leq$100mm). Das Ergebnis der Untersuchung ist in Abb. 7.9 dargestellt. Im Stillstand des Objektes (**A**, **B**) sieht man die gedämpften Schwingungen des Roboters, was durch die dynamischen Eigenschaften des Gesamtsystems der optisch geführten Robotersteuerung verursacht wird.

Um bei schnellen Geschwindigkeiten den Bereich vernünftiger Verfolgung zu erweitern, wurde die Konstante des P- Reglers vergrößert und auf *k=1.2* gesetzt. Dies führt als Folge zur Vergrößerung des Objektbewegungssignals $\Delta\vec{P}_o$ in der Rückführung der optisch geführten Robotersteuerung (siehe Abb. 2.3). Die Abbildung 7.10 zeigt das Ergebnis dieser Veränderung der Reglerkonstante bei der Objektverfolgung. Daraus ist ersichtlich, dass *k*-Vergrößerung zur Erweiterung der Objektverfolgungsbereichs führt und damit die oben beschriebene Vermutung bzgl. der stetigen Vergrößerung des Fehlers bei höheren

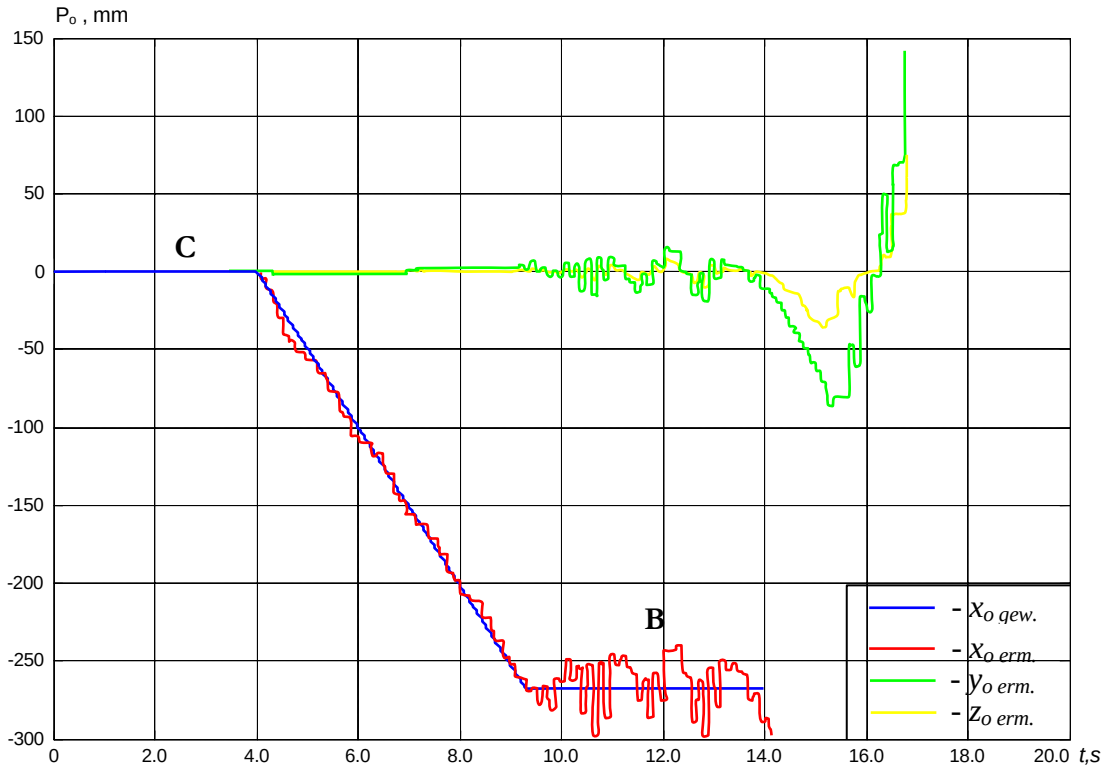


Abbildung 7.8: Der gewünschte und der visuell ermittelte Bewegungsverlauf des Objektes entlang der x- Achse mit den für den Experiment 1 vorgegebenen Geschwindigkeiten und mit der Vorgabe der halben maximalen Geschwindigkeit und Beschleunigung des Roboters

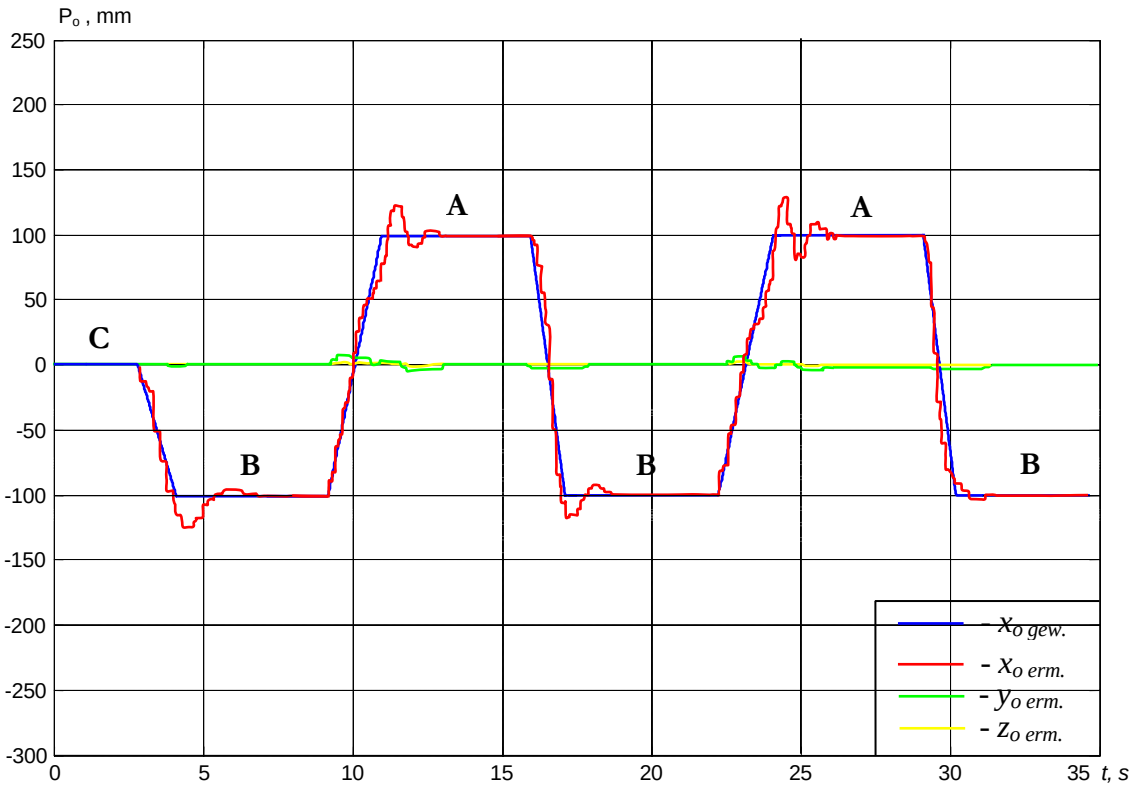


Abbildung 7.9: Der gewünschte und der visuell ermittelte Bewegungsverlauf des Objektes entlang der x- Achse mit den für den Experiment 2 vorgegebenen Geschwindigkeiten

Geschwindigkeiten bestätigt wird. Allerdings kann bei der Geschwindigkeit 200mm/s die Objektverfolgung nicht mehr gewährleistet werden, da das Gesamtsystem der Steuerung instabil wird. Die weitere Erhöhung der Konstante des P- Reglers führt zu keinen besseren Ergebnissen, sondern zu stärkeren Schwingungen des Roboters im Stillstand des Objektes.

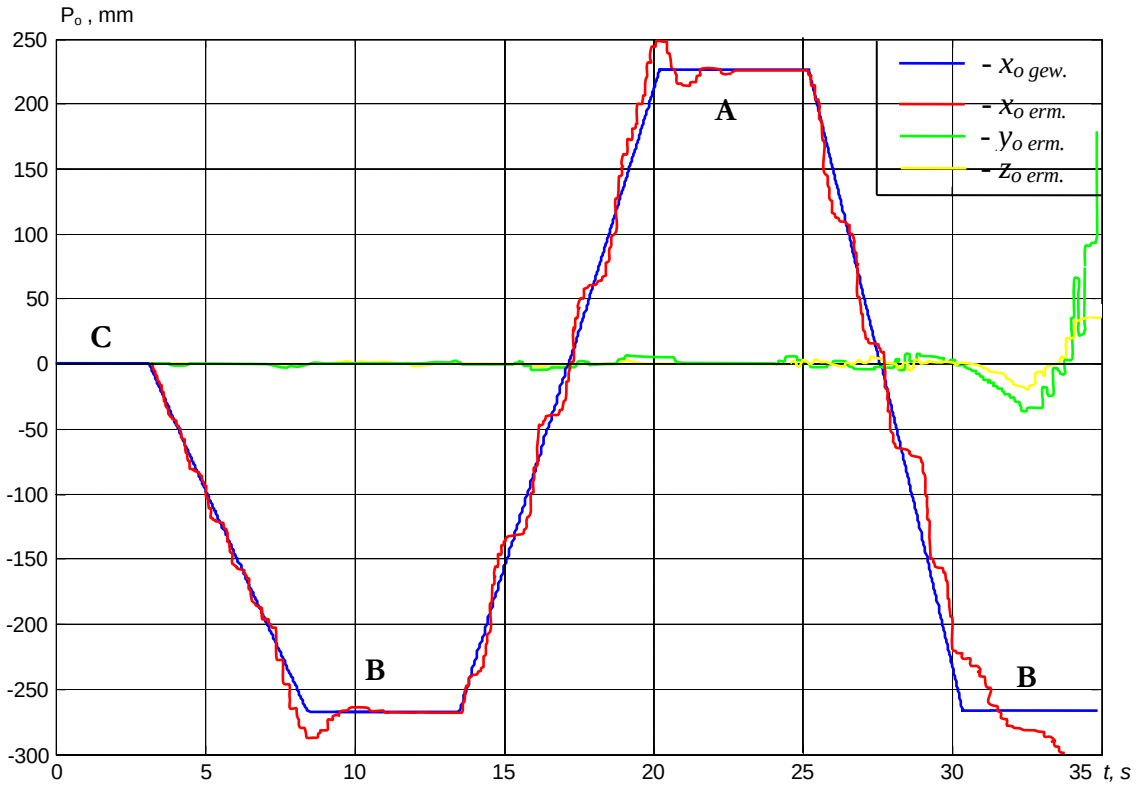


Abbildung 7.10: Der gewünschte und der visuell ermittelte Bewegungsverlauf des Objektes entlang der x-Achse mit den für den Experiment 2 vorgegebenen Geschwindigkeiten und k=1.2 des P- Reglers

7.3 Zusammenfassung

Die Testergebnisse der experimentellen Verifizierung des optisch geführten Systems der Robotersteuerung zeigen, dass die untersuchte Methode für die Objektverfolgung erfolgreich eingesetzt werden kann. Dabei sind die dynamischen Eigenschaften der Objektverfolgung erwartungsgemäß im wesentlichen von der Schnelligkeit der Bestimmung der Lage des Objektes im Kameramodell abhängig. Die anderen Komponenten der optisch geführten Robotersteuerung können mit der Abtastzeit von 1ms im Online- Betrieb (auch z.B. inverse Kinematik) arbeiten. Die Abtastzeit des Kameramodells beträgt dagegen mit der in dieser Arbeit verfügbaren Hardware mindestens 100ms. Die Minimierung dieser Zeit ist von großer Bedeutung, weil diese bei dem vorgeschlagenen Konzept der optisch geführten Robotersteuerung als reine Totzeit in eine rückgeführte Struktur eingeht. Bei deren wesentlicher Reduzierung sind schnellere Objektbewegungen realisierbar. Eine etwas detailliertere Analyse der Ergebnisse zeigt, dass die maximale Geschwindigkeit der Objektverfolgung, die auf den langen Distanzen erreicht werden konnte, 100mm/s beträgt. Die höheren Verfolgungsgeschwindigkeiten (bis zu 200mm/s) sind möglich, allerdings nur auf relativ kurzen Strecken (etwa 200mm). Die Berechnung der Lage des Endeffektors aufgrund

der visuellen Information erfolgt im Kameramodell mit der Abtastzeit von 100ms, da 4 Laserdioden mit einer Periode von 20ms nacheinander ein- und ausgeschaltet werden. Die Reduzierung dieser Zeit kann nicht ohne Verlust der nützlichen Leuchtinformation der Dioden erfolgen und als Folge können Kameras die modulierten Signale nicht mehr voneinander unterscheiden. Die Reduzierung der Abtastzeit des Kameramodells kann durch Verwendung mehrfarbiger Dioden und spezieller PSD-Chips erfolgen. Diese PSD-Chips bestimmen durch die ausgestrahlte Wellenlänge der einzelnen Leuchtdioden deren genaue Position. Die Lichtmodulation fällt dabei völlig aus. Die Berechnung der Endeffektorlage aus der visuellen Information im Kameramodell könnte dann unter Nutzung moderner Mehrprozessorsysteme in einer Zykluszeit von 1ms erfolgen.
Da für viele Anwendungen in der Industrie und in der Medizin langsame Bewegungen bei der Objektverfolgung erforderlich sind (z.B. für Klebe- Schweißaufträge, Operationen), reichen Geschwindigkeiten unter 100mm/s aus. Dabei muss aber die Genauigkeit der Objektverfolgung für den praktischen Einsatz in der Industrie verbessert werden. Dies kann durch Unterdrückung der Störsignale der Kameras mit Hilfe dazu geeigneter Filter erfolgen. Dabei kann die Geschwindigkeit und die Beschleunigung des Roboters bis zum maximalen Wert erhöht werden und dies wird letztendlich zur Erhöhung der Objektverfolgungsgeschwindigkeit führen. Die dynamischen Eigenschaften des Konzeptes können auch durch die Wahl eines für die notwendigen Geschwindigkeiten angepassten Reglers verbessert werden.

Anhang A

Aufbau der Software im Signalprozessorsystem

Basierend auf dem Gesamtsystem (siehe Abb. 2.3) wurde zur optisch geführten Robotersteuerung eine Komplettsoftware für den verwendeten Prozessrechner entwickelt. Zur Entwicklung des Softwarepakets für die Robotersteuerung wurden die Simulink-Routinen auf den verschiedenen Prozessoren im Signalprozessorsystem von dSpace programmiert. Die zugehörige dSpace-Software stellt für Matlab/Simulink ein System dar, welches es ermöglicht, die einzelnen Ein/Ausgangskanäle der Interfacekarte bequem durch Simulink-Blöcke anzusprechen sowie Simulink-Modelle direkt in Matlab zu übersetzen und auf die dSpace-Karte zu laden.
In der Abbildung A.1 ist der Zusammenhang aller Hauptsoftwaremodule entsprechend der Prozessororganisation im Signalprozessorsystem dargestellt. Dabei sind auch die Verbindungen zwischen Softwaremodulen (Prozessoren) durch Datenaustausch dargestellt. Die auf dem Prozessor ***master*** laufenden Softwaremodule dienen dem Datenaustausch zwischen dem Roboter und dem Signalprozessorsystem. In diesem Softwaremodul sind außerdem Nebenfunktionen zur Erzeugung der Überwachungssignale für die verschiedenen Betriebsarten der Robotersteuerung enthalten.

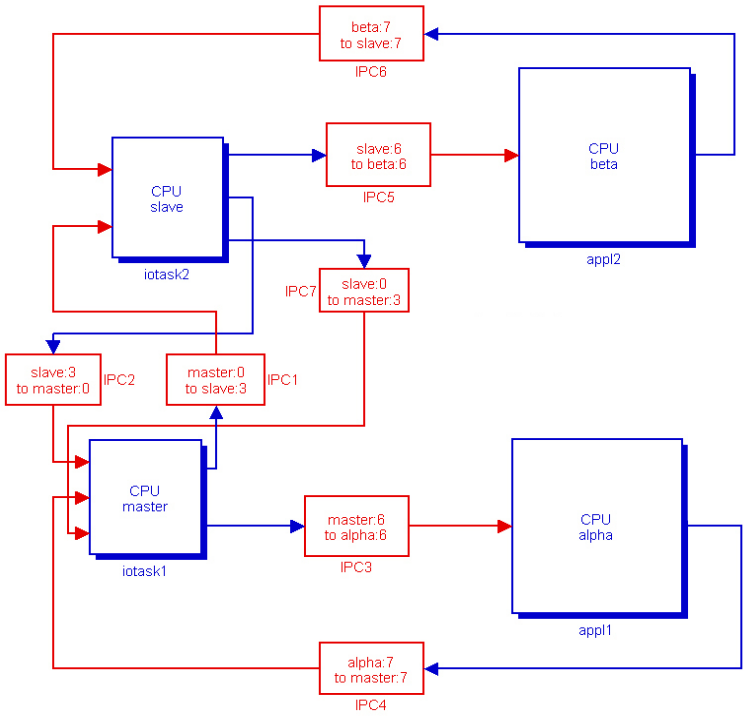


Abbildung A.1: Softwarestruktur im Signalprozessorsystem

Die Modulation der Laserdiodensignale sowie auch die Verarbeitung der gemessenen visuellen Daten der beiden Kameras sind auf dem Prozessor ***slave*** realisiert.
Auf dem leistungsstarken Prozessor ***alpha*** sind die entwickelten Programme zur Berechnung der inversen Kinematik „LMoptim77.c“ und der Lage (Position und Orientierung) des Objektes mittels der visuellen Information der beiden Kameras „visual.c“, und zur Bestimmung der inneren „ikp.c“ und äußeren Kameraparameter „okp.c“ sowie auch für die Trajektoriengenerierung „trajgen.c“ implementiert.
Auf dem Prozessor ***beta*** ist keine Software implementiert. Im Anschluss sind alle diese Hauptsoftwaremodule in der Abbildungen A.2, A.3, A.4 grafisch dargestellt.
Mittels des dSpace *Control Desk* Programms wurde auf dem Rechner schlussendlich eine graphische Oberfläche aufgebaut, die die Signale visualisiert und die Parameter des laufenden Echtzeitprozesses zu variieren erlaubt (siehe Abb. A.5). Dabei können verschiedene Betriebsarten durch entsprechende Auswahltasten auf der grafischen Oberfläche ausgewählt werden. Zurzeit sind fünf Betriebsarten für die Robotersteuerung möglich: *Hand*, *Online*, *IKP*, *OKP* und *VB*, die weiter unten beschrieben werden.
Nach dem Aufruf der grafischen Oberfläche können die Befehle *Reset*, *Init* und *Halt* an alle Achsen des Roboters gesendet werden. Mit dem Befehl *Init* sucht der Roboter seine Initialposition. Im *Halt*-Modus löst man den Nothalt aus, der Roboter nimmt in diesem Zustand keine Befehle mehr an. Dieser Modus kann durch den *Reset*-Befehl aufgehoben werden.
Die für alle Roboterachsen maximale Geschwindigkeit und Beschleunigung entsprechend der Trajektoriengenerierung nach Kapitel 4 kann durch Betätigung der prozentuellen Vorgabetasten *Vel(%)* und *Acc(%)* eingestellt werden. Dabei können die Befehle *Set Acc*, *Set Vel* und *RAMP* im *Programm-Mode* ausgewählt werden, wobei die *Set Acc* -Taste zur Beschleunigungsvorgabe, *Set Vel* -Taste zur Geschwindigkeitsvorgabe und *RAMP* -Taste zur Nullpositionsvorgabe (alle Achswinkel des Roboters gleich null) dienen. Im Anschluss hieran können die Betriebsarten des Roboters ausgewählt werden.
In der Betriebsart *Hand* können alle Roboterachsen durch Betätigung entsprechenden Tasten gesteuert werden. Unter der Betriebsart *Online* wird die Berechnung der inversen Kinematik in Echtzeit durchgeführt. Dabei kann jede einzelne Position (*X*, *Y*, *Z)* bzw. Orientierung (α, β, γ) des Endeffektors in Echtzeit durch die Betätigung der Zunahme- und Abnahmetasten „+/–“ gesteuert werden.
Die restlichen drei Betriebsarten arbeiten unter der Betriebsart *Online*. Die Betriebsart *IKP* wird durch die Betätigung der Taste „on“ eingeschaltet und mit Taste „off“ beendet. Unter dieser Betriebsart *IKP* wird die Berechnung der inneren Kameraparameter durchgeführt. Nachdem die Betriebsart *OKP* genau wie bei der Betriebsart *IKP* mit der Taste „on“ eingeschaltet wird, werden die äußeren Kameraparameter berechnet. Nach Ausschaltung der Betriebsart OKP mit der Taste „off“ kann man zur Betriebsart *VB* übergehen. In der Betriebsart *VB* werden die Positionen und die Orientierungen des Objektes aufgrund der visuellen Information beider Kameras berechnet und die entsprechende Trajektorien zur Verfolgung des Objektes für den Endeffektor des Roboters generiert.
Im oberen Teil der grafischen Oberfläche (siehe Abb. A.5) sind die vier Laserdioden aus Sicht der beiden Kameras grafisch dargestellt. Die zwei linken Grafiken zeigen die Signale am Ausgang des Positionsmessverstärkers (X, Y, Summensignal) der Kameras. Die zwei rechten Grafiken zeigen dagegen der korrigierte Ausgangsignal des Kameraverstärkers, bei der die Wirkung des äußeren Lichtes der Umgebung des Roboters unterdrückt wird.

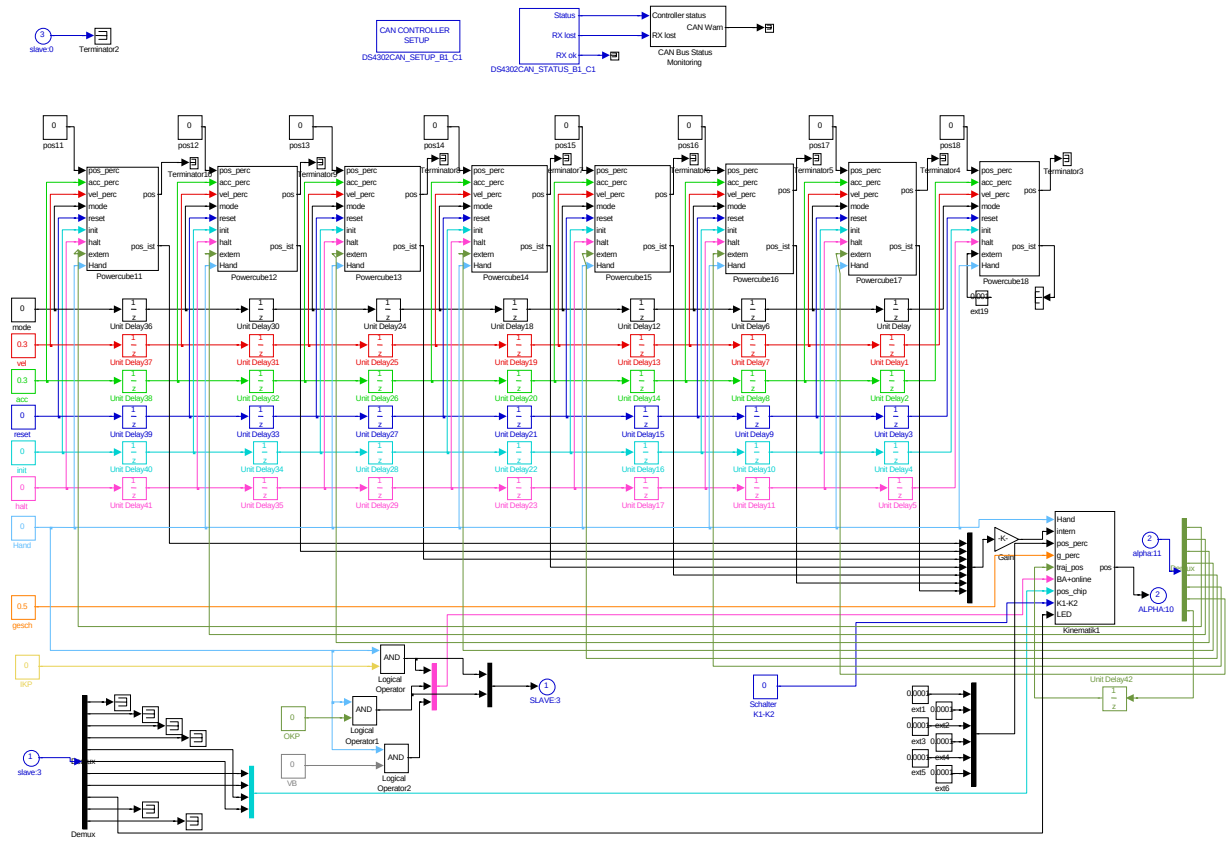


*Abbildung A.2: Softwarestruktur auf dem **master**- Prozessor*

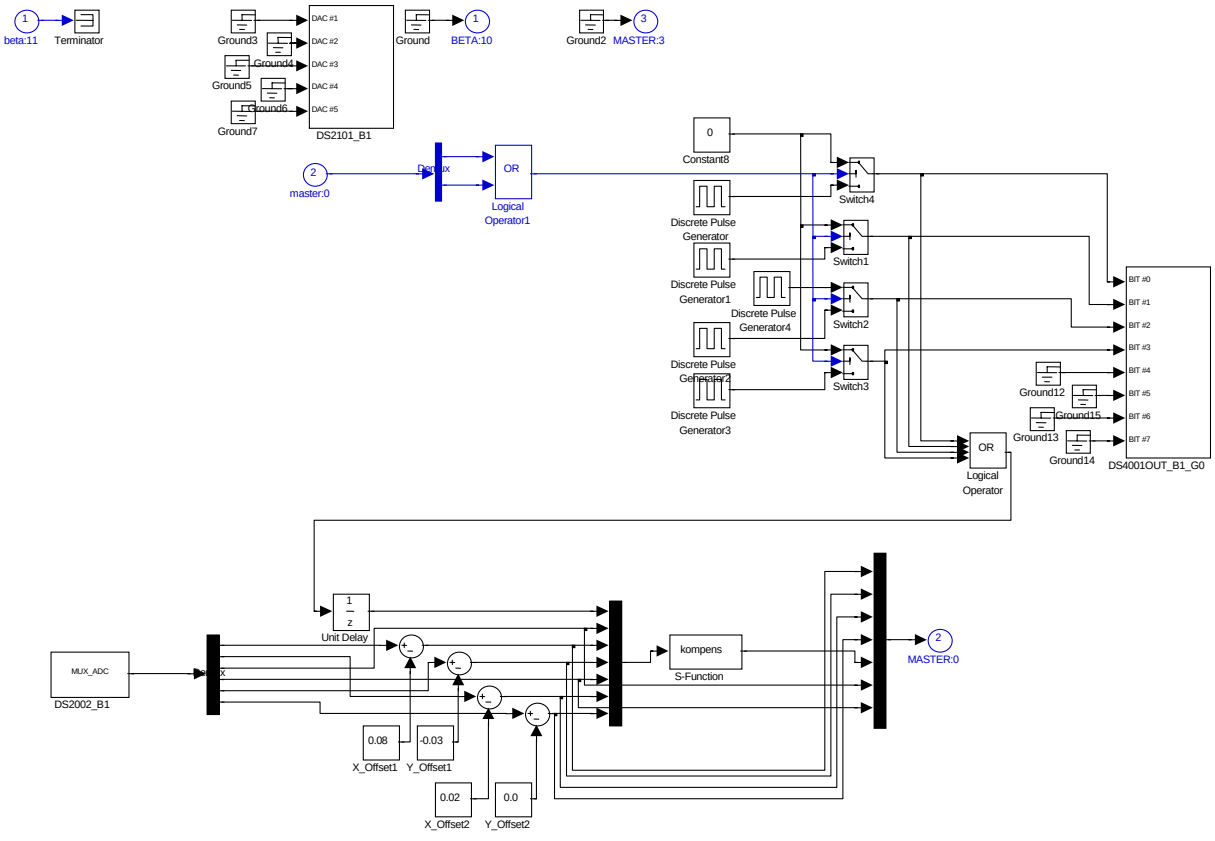


*Abbildung A.3: Softwarestruktur auf dem **slave**- Prozessor*

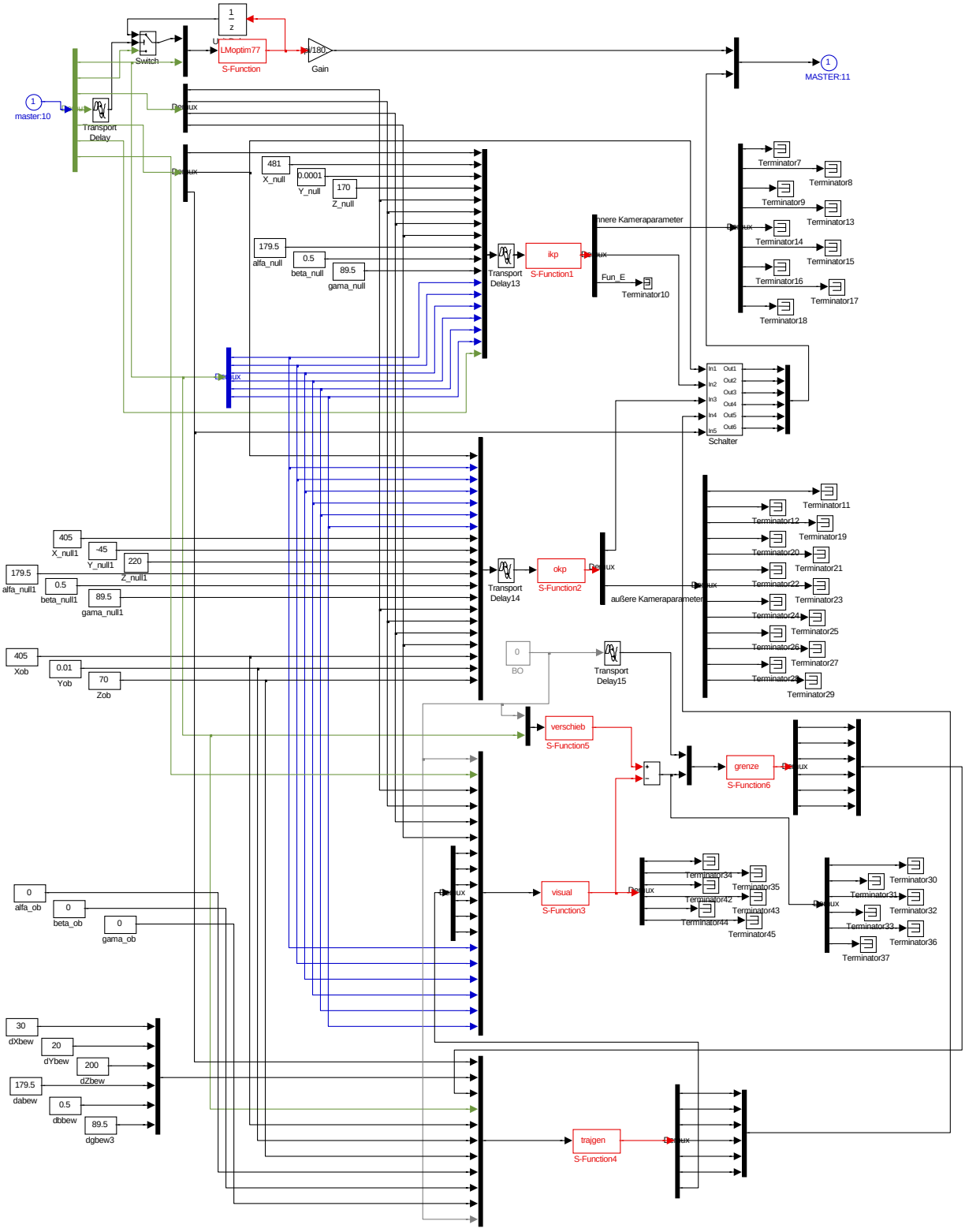


Abbildung A.4: Softwarestruktur auf dem ***alpha****- Prozessor*

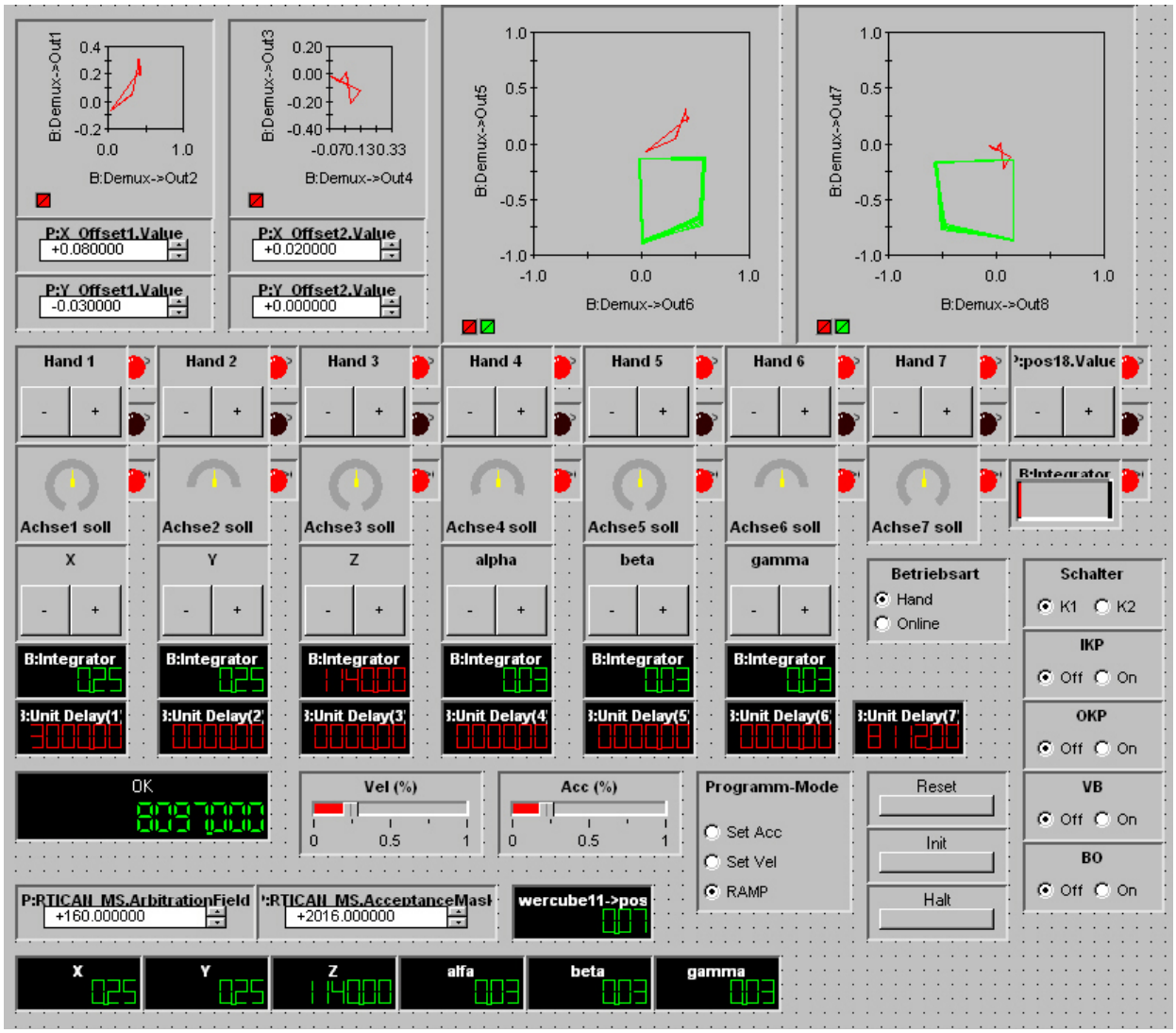


Abbildung A.5: Grafische Oberfläche der Robotersteuerung

Literaturverzeichnis

[1] J. S. Albus. Brains, *behavior and robotics*, Byte Books. 1981.

[2] P. K. Allen, A. Timcenko, B. Yoshimi and P. Michelman. *Automated Tracking and Grasping of a Moving Object with a Robotic Hand-Eye System*. IEEE Transactions on Robotics and Automation, vol.9, no.2, 1993, pp. 152-165.

[3] R. L. Anderson. *Real time intelligent visual control a robot*. Proc. of IEEE Int. Workshop on intelligent Control, pp. 89-94, 1985

[4] E. Appleton. *Industrieroboter: Anwendungen*. Weinheim, 1991.

[5] W. J. P. Barnes and M.H. Gladden. *Feedback and Motor Control in Invertebrates and Vertebrates*. Croom Helm, London 1985.

[6] Eugene Hecht. *Optik*. Addison-Wesley, 1989.

[7] G. M. Bone and M. A. Elbestawi. *Sensing and control for automated robotic edge deburring*. IEEE Trans. on Industrial Electronics, vol.41, no.2, pp.137-145, 1994.

[8] M. Bowman and A. Forrest. *Visual detection of differential movement: Applications to robotics*. Robotica, 6:7-12, 1988.

[9] M. Brady. *Robot Motion: Basics of robot motion planning and control*. The MIT Press, Cambridge, Massachusetts, 1. Aufl., 1993.

[10] R. Brause. *Performance and storage requirements of topology-conserving maps for robot manipulator control*. Technikal Report 5/89, Universität Frankfurt, FB Informatik, 1989.

[11] I. N. Bronstein und K. A. Semendjajew. *Taschenbuch der Mathematik*. B. G. Teubner, Stuttgart, 25. Auflage, 1991.

[12] R. Bukowski. *Robot hand-eye coordination rapid prototyping environment*. In Proc. ISIR, pages 16.15-16.28, October 1991.

[13] G. Buttazzo, B. Allotta and F. Fanizza. *Mousebuster: a robot system for catching fast moving objects by vision*. In Proc. IEEE Int. Conf. Robotics and Automation, pages 932-937, 1993.

[14] C. Canudas de Wit, B. Siciliano and G. Bastin. *Theory of Robot Control*. Springer, 1996

[15] F. Chaumette, P. Rives and B. Espiau. *Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing*. In Proc. IEEE Int. Conf. Robotics and Automation, pages 2248-2253, 1991.

[16] C. Chen and M. M. Trivedi. *Task Planning and Action Coordination in Integrated Sensor-Based Robots*. IEEE Trans. on Systems, Man, and Cybernetics, vol. 25, no. 4, April, 1995.

[17] W. Chen and B. C. Jiang. *3-D camera calibration using vanishing point concept*. Pattern Recognition, 24(1), pp. 57-67, 1991.

[18] R. N. Chiou, K. C. Hung, C. H. Chen and J. Y. Lee. *Image reconstruction and stereo matching for polyhedron objects*. The 7th Scandinavian Conference on Image Analysis, Denmark, August 1991, pp. 218-224.

[19] P. I. Corke. *Visual Control of Robot Manipulators- A Review*. In K. Hashimoto, editor, Visual Servoing, pp. 1-32, World Scientific. 1994.

[20] P. I. Corke. *Visual Control of Robots*. Research Studies press Ltd., 1996.

[21] P. I. Corke and M. C. Good. *Dynamic Effects in Visual Closed-Loop Systems*. IEEE Trans. on Robotics and Automation, vol. 12, no.5, pp.671-683, 1996.

[22] P. Y. Coulon and M. Nougaret. *Use of a TV camera system in closed-loop position control of mechanisms*. In A. Pugh, editor, International trends in manufacturing technology, ROBOT VISION, pages 117-127, IFS Publications, 1983.

[23] John J. Craig. *Introduction to Robotics– Mechanics and Control*. AddisonWesley, Reading, MA, 2. Auflage, 1989.

[24] A. Cretual and F. Chaumette. *Image-based visual servoing by integration of dynamic measurements*. IEEE International Conference on Robotics and Automation, Leuven, May 1998, pp. 1994-2001.

[25] J. L. Eklund, C. Svensson and A. Astrom. *VLSI implementation of a focal plane image processor -a realization of the near-sensor image processing concept*. IEEE Trans. VLSI Systems, vol. 4, no. 3, pp.322-335, 1996.

[26] H. Fässler, H. Beyer and J. Wen. *A robot ping pong player: optimized mechanics, high performance 3D vision, and intelligent sensor control*. Robotersysteme, 6:161-170, 1990.

[27] J. T. Feddema, C. S. G. Lee and O. R. Mitchell. *Weighted selection of image features for resolved rate visual feedback control*. IEEE Trans. Robot. Autom., 7(1); pp. 31-47, February 1991.

[28] K. S. Fu, R. C. Gonzalez and C. S. G. Lee. *Robotics: Control, Sensing, Vision and Intelligence*. New York: McGraw-Hill, 1987.

[29] J. F. Gardner, A. Brandt and G. Luecke. *Applications of neural networks for coordinate transformations in robotics*. Journal of Intelligent ans Robotic Systems, 8, pp. 361-373, 1993

[30] J. O. Gray and D. G. Galdwell. *Advanced Robotics and Intelligent Machines*. Published by the Institution of Electrical Engineers, London, United Kingdom, 1996.

[31] E. Grosso, G. Metta, A. Oddera and G. Sandini. *Robust Visual Servoing in 3-D Reaching Tasks*. IEEE Trans. on Robotics and Automation, vol.12, no.5, pp.732-742, 1996.

[32] A. Guez und Z. Ahmad. *Solution to the inverse kinematics problem in robotics by neuronal networks*. In IJCNN'88, San Diego, volume II, pp. 617-624, 1988

[33] A. Guez und Z. Ahmad. *Accelerated convergence in the inverse kinematics via multilayer feedforward networks*. In IJCNN'89, Washington, volume II, pp. 341-344, 1989

[34] W. A. Gruver. *Intelligent robotics in manufacturing, service, and rehabilitation: an overview*. IEEE Trans. On Industrial Electronics, vol. 41, no. 1, pp. 4-11, 1994

[35] G. D. Hager, G. Grunwald and G. Hirzinger. *Feature-based visual servoing and its application to Telerobotics*. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, pp.164-171, 1994.

[36] G. D. Hager, W. C. Chung and A. S. Morse. *Robot Hand-Eye Coordination Based on Stereo Vision*. IEEE Control System, Feb. 1995, pp. 30-39.

[37] J. Hakala. *Bedarfsangepasste neuronale Netze mit topologischer Kodierung zur Bestimmung der inversen Kinematik von Gelenkrobotern*. Dissertation. Friedrich- Wilhelms- Universität Bonn, November 1994.

[38] K. Hashimoto and T. Noritsugu. *Visual Servoing with Linearized Observer*. IEEE Int. Conf. on Robotics and Automation, Detroit, pp.263-268, 1999.

[39] J. Hill and W. T. Park. *Real time control of a robot with a mobile camera*. In Proc. 9th ISIR, pages 233-246, Washington, DC, March 1979.

[40] B. K. P. *Horn. Robot Vision*. The MIT Press McGraw-Hill Book Company, 1991.

[41] S. H. Huang and H. -C. Zhang. *Artificial neural networks in manufacturing: concepts, applications, and perspectives*. IEEE Trans. on Components, Packaging and Manufacturing Technology-Part A, vol.17, no.2, pp. 212-228, 1994.

[42] S. Hutchinson, G. Hager and P. Corke. *A tutorial on visual servo control*. IEEE Trans. on Robotics and Automation, vol.12, no.5, pp.651-670.

[43] M. Jägersand, O. Fuentes and R. Nelson. *Experimental evaluation of uncalibrated visual servoing for precision manipulation*. In Proc. IEEE Int. Conf. Robotics and Automations, 1996.

[44] W. Jang, K. Kim, M. Chung and Z. Bien. *Concepts of augmented image space and transformed feature space for efficient visual servoing of an eye-in-hand robot*. Robotica, 9: 203-212, 1991.

[45] W. Jang and Z. Bien. *Feature-based visual servoing of an eye-in-hand robot with improved tracking performance*. In Proc. IEEE Int. Conf. Robotics and Automation, pages 2254-2260, 1991.

[46] R. Kelly. *Robust asymptotically stable visual servoing of planar robots*. IEEE Trans. on Robotics and Automation, vol.12, no.5, pp.759-766, 1996.

[47] R. Kelly, F. Reyes, J. Moreno and S. Hutchinson. *A Two Loops Direct Visual Control of Direct- Drive Planar Robots with Moving Target*. IEEE Int. Conf. on Robotics and Automation, Detroit, pp.599-604, 1999.

[48] T. Komuro, I. Ishii and M. Ishikawa. *Architecture using general purpose processing elements for 1ms vision system*. Proc. CAMP'97, pp. 276-279, 1997.

[49] P. Kosmol. *Methoden zur numerischen Behandlung nichtlinearer Gleichungen und Optimierungsaufgaben*. B. G. Teubner Verlag, Stuttgart, 1989.

[50] W. Krabs. *Einführung in die lineare und nichtlineare Optimierung für Ingenieure*. Leipzig, 1983

[51] M. Kuperstein. *INFANT neural controller for adaptive neural controller for sensory-motor coordination*. Neural Networks, vol.4, 1991, pp. 131-144.

[52] S. Lee. *A self-calibration model for hand-eye systems with motion estimation*. Math. Comput. Model (UK), vol.24, no.5-6, pp. 49-77, 1996.

[53] J. S. Lee, Il H. Suh, B. J. You, S. R. Oh. *A Novel Visual Servoing Approach involving Disturbance Observer*. IEEE Int. Conf. on Robotics and Automation, Detroit, pp. 269-274, 1999.

[54] Lenz, Reimar K. und Roger Y. Tsai. *Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 10(5):713–720, September 1988.

[55] M. X. Li and D. Betsis. *Head-Eye Calibration, Proceedings*. Fifth International Conference on Computer Vision, Cambridge, MA, USA, 20-23, pp. 40-45, June 1995.

[56] Y. F. Li and M. H. Lee. Applying Vision Guidance in Robotic Food Handling. IEEE Trans. On Robotics and Automation, vol.3, no.1, pp. 4-12, 1996.

[57] J. Lupo. *DARPA: Neural Network Study*. AFCEA International Press, Burke, USA, 1988.

[58] A. G. Makhlin. *Stability and sensitivity of servo vision systems*. Proc. 5th Int Conf on Robot Vision and Sensory Controls - RoViSeC 5, pages 79-89, October 1985.

[59] D. Marr and T. Poggio. *A theory of human stereo vision*. Proc. Roy. Soc. London, vol. B 204, pp. 301-328, 1979.

[60] B. Mel. *Connectionist Robot Motion Planning*. Academic Press 1990.

[61] W. T. Miller. *Sensor-based control of robotic manipulators using a general learning algorithm*. IEEE Journal of Robotics and Automation, vol.3, no.2, 1987, pp. 157-165.

[62] W. T. Miller. *Real-time application of neural networks for sensor-based control of robots with vision*. IEEE Trans. Systems, Man, and Cybernetics, vol. 19, no.4, 1989, pp. 825-831.

[63] V. Mut, O. Nasisi, R. Carelli and B. Kuchen. *Tracking Adaptive Impedance Robot Control With Visual Feedback*. IEEE Int. Conf. on Robotics and Automation, Leuven, pp. 2002-2007, 1998.

[64] Y. Nakabo and M. Ishikawa. *Visual Impedance Using 1ms Visual Feedback System*. Proceeding of IEEE International Conference on Robotics and Automation, Leuven, pp. 2333-2338, 1998.

[65] B. J. Nelson, N. P. Papanikolopoulos and P. Khosla. *Robotic visual servoing and robotic assembly tasks*. IEEE Trans. on Robotics and Automation, vol.3, no.2, pp.23-31, 1996.

[66] Dragomir N. Nenchev. *Redundancy Resolution through Local Optimization: A Review*. Journal of Robotic Systems, 6(6):769–798, 1989.

[67] N. Papanikolopoulos, P. K. Khosla and T. Kanade. *Vision and control techniques for robotic visual tracking*. IEEE Int. Conf. Robotics and Automation, pp. 857-864, 1991.

[68] N. Papanikolopoulos, P. K. Khosla. *Adaptive robotic visual tracking: theory and experiments*. IEEE Trans. Automatic Control, vol. 38, no. 3, pp. 429-445, 1993.

[69] Jonghoon Park, Wan-Kyun Chung and Youngil Youm. *Characteristics of Optimal Solutions in Kinematic Resolution of Redundancy*. IEEE Transactions on Robotics and Automation, 12(3):471–478, Juni 1996.

[70] R. P. Paul. *Robot Manipulator: Mathematics, Programming and Control*. MIT Press, Cambridge, Mass.

[71] R. P. Paul and H. Zhang. *Design of a robot force / motion server*. In Proc. IEEE Int. Conf. Robotics and Automation, volume 3, pages 1878-83, Washington, USA, 1986.

[72] E. M. Petriu. *Multisensor system for mobile robot navigation*. IEEE Instrumentation and Measurement Technology Conference and IMEKO Technical Committee 7. Conference Proceedings, Brussels, Belgium, 4-6, pp. 388-392, vol.1, June 1996.

[73] J. A. Piepmeier, G. V. Mcmurray and H. Lipkin. *Tracking a Moving Target with Model Independent Visual Servoing: a Predictive Estimation Approach*. IEEE International Conference on Robotics and Automation, Leuven, 16-20, pp. 2652-2657, May 1998.

[74] Pyung H. Chang. *A Closed-Form Solution for Inverse Kinematics of Robot Manipulators with Redundancy*. IEEE Journal of Robotics and Automation, RA–3(5):393–403, 1987.

[75] C. Rosen. *Machine intelligence research applied to industrial automation*. Sixth report. Technical report, SRI International, 1976.

[76] C. Rosen. *Machine intelligence research applied to industrial automation*. Eighth report. Technical report, SRI International, 1978.

[77] C. Samson, B. Espiau and M. L. Borgne. *Robot Control: the Task Function Approach*. Oxford University Press, 1990.

[78] R. Schmid. *Industrielle Bildverarbeitung*. Verlag Vieweg, 1995.

[79] N. Schneider. *Kantenhervorhebung und Kantenverfolgung in der industriellen Bildverarbeitung*. Verlag Vieweg, 1994.

[80] M. Schulz. *Geometrische Kalibration von CCD- Kameras*. Ruprecht- Karls- Universität Heidelberg, Diplomarbeit, 1997.

[81] P. Sharkey, I. Reid, P. McLauchlan and D. Murray. *Real-time control of a ractive stereo head/eye platform*. Proc. 29th CDC, pages CO.1.2.1-CO.1.2.5, 1992.

[82] Y. Shirai and H. Inoue. *Guiding a robot by visual feedback in assembling tasks*. Pattern Recognition, 5:99-108, 1973.

[83] S. K. Sim, M. Y. Teo. *Enhancing flexibility of vision-based robots using artificial neural network approach*. Proceedings of the 3rd International Conference, Computer Integrated Manufacturing, Singapore, 11-14, pp. 1479-1490, vol.2, July 1995.

[84] S. Skaar, W. Brockman and R. Hanson. *Camera-space manipulation*. Int. J. Robot. Res., 6(4):20-32, 1987.

[85] P. van der Smagt. *A Monocular Robot Arm Can Be Neurally Positioned*. Int. Conference on Intelligent Autonomous Systems, 1995.

[86] K. Stanley, Q. M. J. Wu, A. Jerbi and W. A. Gruver. *Neural Network-Based Vision Guided Robotics*. IEEE Int. Conf. on Robotics and Automation, Detroit, pp. 281-286, 1999.

[87] J. R. Stenstrom and C. I. Connolly. *Constructing object models from multiple images*. Int. Journal of Computer Vision, 9(3), pp.185-212, 1992.

[88] Liping Sun. *Beitrag zur Entwicklung lernfähiger Strukturen für die visuomotorische Koordination von Robotern*. Universität Magdeburg, Iinstitut für Automatisierungstechnik, Dissertation, 2000

[89] Y. W. Sung, D. K. Cho and M. J. Chung. *A Constrained Optimization Approach to Resolving Manipulator Redundancy*. Journal of Robotic Systems, 13(5):275–288, 1996.

[90] Tsai, Y. Roger. *A Versatile Camera Calibration Technique for High-Accuracy 3D Machine Vision Metrology Using Off-the-Shelf TV Cameras and Lenses*. IEEE Journal of Robotics and Automation, pp. 323–344, August 1987.

[91] J. Urban, G. Motyl and J. Gallice. *Real-time visual servoing using controlled illumination*. Int. J. Robot. Res., 13(1):93-100, February 1994.

[92] S. Venkatesan and C. Archibald. *Realtime tracking in five degrees of freedom using two wrist-mounted laser range finders*. In Proc. IEEE Int. Conf. Robotics and Automation, pages 2003-2010, 1990.

[93] D. Vernon and M. Tistarelli. *Using camera motion to estimate range for robotic parts manipulation*. IEEE Trans. Robot. Autom., 6(5):509-521, October 1990.

[94] J. A. Walter und K. J. Schulten. *Implementation of self-organising neuronal networks for visuo-motor control of an industrial robot*. IN IEEE Trans. On Neuronal Networks, pp. 86-95, 1993.

[95] L. Weiss. *Dynamic Visual Servo Control of Robots: an Adaptive Image-Based Approach*. PhD thesis, Carnegie-Mellon University, 1984.

[96] S. Wijesoma, D. Wolfe and R. Richards. *Eye-to-Hand Coordination for Vision-Guided Robot Control Applications*. International Journal of Robotics Research, 12(1), pp. 65-78, 1993.

[97] W. J. Wilson, C. C. W. Julls and G. S. Bell. *Relative end-effector control using cartesian position based visual servoing*. IEEE Trans. on Robotics and Automation, vol.12, no.5, pp.684-696, 1996.

[98] J. Woodfill and B. V. Herzen. *Real-time stereo vision on the parts reconfigurable computer*. IEEE Workshop on FPGAs for Custom Computing Machines, pp.242-250, 1997.

[99] D.-Y. Yeung and G. A. Bekey. *Using a context- sensitive learning network for robot arm conrol*. In Proc. IEEE Int. Conf. on Robotics and Automation, pp. 1441-1447, 1989.

[100] D.-Y. Yeung und G. A. Bekey. *On reducing learning time in context- dependent mappings*. IEEE Trans. On Neuronal Networks, 4(1): pp. 31-42, 1993.

[101] Yoshihiko Nakamura. *Advanced Robotics– Redundancy and Optimization*. Addison Wesley, Reading, MA, 1991.

[102] A. M. S. Zalzala, A. S. Morris. *Neural Networks for Robotic Control: Theory and Applications*. Ellis Horwood, 1996.

[103] H. Zhuang and Z. S. Roth. *Camera-Aided Robot Calibration*. CRC Press, Inc. 1996.

Lebenslauf

Persönliche Angaben

Name:	Alexander Krasnych
Wohnort:	70372 Stuttgart
Strasse:	Aberlin-Jörg Str. 5
Geburtstag und –ort:	07.06.1975 in Leninogorsk (Kasachstan)
Staatsangehörigkeit:	deutsch
Familienstand:	ledig

Schulausbildung

09/1982 - 07/1992	Mittelschule Gorlowka (Ukraine) Abschluss: Abitur entsprechender

Studium

09/1992 – 07/1997	Elektrotechnik-Studium / Technische Universität Donezk (Ukraine) Deutsche technische Fakultät Fachrichtung: Elektroantriebs- und Automatisierungstechnik
07/1999	Abschluss: Diplom des Elektroingenieurs
09/1997 – 07/1999	Magisterstudium / TU Donezk, Deutsche technische Fakultät
07/1999	Abschluss: Magisterdiplom
01/2000 – 01/2005	Promotion / Otto-von-Guericke-Universität-Magdeburg Thema: „Optisch geführtes Steuerungskonzept eines Roboters in Echtzeit“.

Praktika

01/1997 – 05/1997	Vordiplompraktikum am Institut für Automatisierungstechnik, Otto-von-Guericke-Universität Magdeburg
03/1998 – 09/1998	Betriebspraktikum im Unternehmen SIEMENS AG, Stuttgart
10/1998 – 02/1999	Erarbeitung der Magisterdissertation am Institut für Automatisierungstechnik, Universität Magdeburg

Magdeburg, 20.07.2005