Adaptiv vorausplanende Steuerung für schnelle sensorbasierte Roboterbewegungen

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

der Fakultät für Informatik

der Universität Karlsruhe (Technische Hochschule)

vorgelegte

Dissertation

von

Dipl.-Ing. Friedrich Lange

aus Kassel

Tag der mündlichen Prüfung	3.Februar 2003
Erster Gutachter	Prof. Dr.-Ing. R. Dillmann
Zweiter Gutachter	Prof. Dr.-Ing. G. Hirzinger

Übersicht

Herkömmliche Robotersteuerungen, wie sie in der industriellen Fertigung eingesetzt werden, bewirken bei hohen Geschwindigkeiten teilweise erhebliche Bahnabweichungen der Werkzeugspitze, sodass bei kritischen Aufgabenstellungen, die hohe Bahngenauigkeit erfordern, eine Anpassung des Steuerungsprogramms erforderlich wird. In der Regel müssen die überwiegend offline programmierten Bahnen manuell angepasst werden, sofern die Geschwindigkeit des Industrieroboters nicht reduziert werden soll. Diese Fragestellung wird in dieser Arbeit qualitativ und quantitativ untersucht.

Leistungsfähige adaptive Algorithmen zur Kompensation der Bahnabweichungen bei schnellen Roboterbewegungen sind weder im industriellen Bereich noch im wissenschaftlichen Umfeld direkt verfügbar. Gleichzeitig besteht aber ein großer Bedarf, offline programmierte Aufgaben für Industrieroboter sofort fehlerfrei ausführen zu können.

Aus diesem Grund wird in dieser Arbeit ein adaptives Korrekturverfahren zur Reduktion von schnellen Roboterbahnabweichungen entwickelt. Das korrigierte Verhalten wird in Form von Parametern einer nichtkausalen Vorsteuerung der Soll-Bahn abgespeichert. Zur Kompensation von nichtlinearen Streckeneigenschaften werden Neuronale Netze eingesetzt. Die Elastizitäten der Robotergelenke werden dabei durch spezielle Messeinrichtungen erfasst und berücksichtigt.

Der durch das adaptive System geregelte Roboter ist Kern einer Architektur zur Anpassung von schnellen Roboterbahnen an eine a priori unbekannte oder zumindest ungenau bekannte oder zwischenzeitlich veränderte Umgebung. Dabei wird die reale Umgebung online sensorisch erfasst. Die Architektur erlaubt die Fusion von unterschiedlichen Typen von Sensoren zur aufgabenabhängigen Spezifikation der gewünschten Bewegung. Im Gegensatz zu anderen Ansätzen wird die Dynamik des Zielsystems im gesamten Entwurf explizit berücksich-

tigt. Zusammen mit der Vorsteuerung erlaubt dies eine schnelle und fehlerfreie Ausführung sensorisch definierter Bahnen.

Die Experimente mit dem vorgestellten Verfahren zeigen, dass eine einmalige Adaption des Roboterverhaltens den dynamischen Bahnfehler gegenüber der ursprünglichen Robotersteuerung bei schnellen Bewegungen nahezu bahnunabhängig deutlich verkleinern kann. Bei einer problemangepassten Adaption für eine bestimmte definierte Bewegung lässt sich der Bahnfehler weiter reduzieren.

Es wird gezeigt, dass das Verfahren in übliche Robotersteuerungen integriert werden kann.

Vorwort

Diese Arbeit entstand im Rahmen mehrerer Forschungsvorhaben beim Deutschen Zentrum für Luft- und Raumfahrt (DLR) in Oberpfaffenhofen sowie am Institut für Prozessrechentechnik und Robotik (IPR) der Universität Karlsruhe. Die Entwicklung der Methode, die experimentelle Verifikation und die Dokumentation erfolgten am Institut für Robotik und Mechatronik des DLR in Oberpfaffenhofen.

Mein besonderer Dank gilt Herrn Prof. Dr.-Ing. R. Dillmann von der Universität Karlsruhe, der mit seinen Anmerkungen und kritischen Fragen sehr dazu beigetragen hat, die Dissertation voran zu bringen. In gleicher Weise danke ich Herrn Prof. Dr.-Ing. G. Hirzinger vom Deutschen Zentrum für Luft- und Raumfahrt, der die Arbeit initiiert und vor allem während der Experimente intensiv betreut hat. Dabei hat er mir den notwendigen forscherischen Freiraum gelassen, der zum Gelingen der Arbeit führte.

Daneben danke ich meinen Kollegen für die vielen fruchtbaren Diskussionen, Vorschläge, Programmteile und Hardware-Komponenten, die in diese Arbeit eingeflossen sind. Namentlich erwähnen möchte ich Herrn Michael Steinmetz, der mir besonders bei den Experimenten mit Rat und Tat zur Seite stand und Herrn Ralf Koeppe, der mich durch Hinweise und Diskussionen bei der Ausarbeitung unterstützte.

Schließlich möchte ich noch meine Frau Annette erwähnen, die mich immer wieder motivierte und die Arbeit trotz des hohen Zeitaufwands unterstützte.

Oberpfaffenhofen, im August 2002 Friedrich Lange

Inhaltsverzeichnis

Liste der verwendeten Symbole

Es werden nur die wichtigsten Symbole und die Seitenzahl der Beschreibung aufgeführt. Nur einmal vorkommende Variablen werden nur beim Auftreten erwähnt.

Allgemeine Notation

n, F, α	Skalar
$\mathbf{x}, \mathbf{F}, \tau$	Vektor
x_i, τ_j	Element eines Vektors
$\mathbf{A}$	Matrix
$\mathbf{A}^T$	transponierte Matrix
$a_{ij}, a_{i,j}$	Element einer Matrix
$\underline{\mathbf{x}}, \mathbf{\Psi}$	Tensor der Vektoren zu verschiedenen Zeitpunkten
r_{wi}	Element von r_w
$\dot{x}$	zeitliche Ableitung
$\hat{x} = E\{x\}$	Schätzwert, Erwartungswert
a', a^*	a entsprechende aber unterschiedliche Größe
Δ	Differenz

Variablen und Funktionen

a_i	Streckenparameter zur Gewichtung von $y(k-i)$ (Seite 124)
b_i	Streckenparameter zur Gewichtung von $u(k-i)$ (Seite 124)
$D, \mathbf{D}$	Dämpfungsparameter (Seite 118)
$e, \mathbf{e}$	Regeldifferenz, Vektor der Achswerte (Seite 55)
$E, \mathbf{E}$	Elastizitätsparameter, Matrix der kartesischen Elastizität (Seite 118)
$f(\cdots)$	Funktion von (z. B. bei Neuronalen Netzen) (Seite 168)
$\mathbf{F}$	Filter für Adaption (Seite 35)
$\mathbf{F}$	Kontaktkraft, im Sensor gemessen (Seite 17)
g_i	Element der Gewichtsfunktion (Impulsantwort) (Seite 129)
$\mathbf{G}$	Übertragungsfunktion der Strecke (Seite 35)
$\mathbf{G}$	Matrix der Gewichtsfunktionen (Impulsantworten) (Seite 101)
$\mathbf{G}$	(vektorielle) Funktion des Gravitationseinflusses (Seite 17)
$\mathbf{I}$	Einheitsmatrix
$\mathbf{I}_r$	Trägheitsmoment des Rotors (Seite 50)
$\mathbf{I}_a$	Trägheitsmoment des Armes (Seite 50)
J	Gütektriterium (Seite 55) (Indizes ab Seite xvii)
$\mathbf{J}$	Jacobi-Matrix (Seite 17)
k	Abtastzeitpunkt (Seite 20)
$\mathbf{K}$	Kalmanverstärkung (Seite 274)
$\mathbf{K}$	Verstärkungsmatrix (Seite 18) (Indizes ab Seite xvii)
$\mathbf{L}$	Filter für Adaption (Seite 34)
m	Zahl der Achsen (Seite 17)
m	Zahl der Elemente (Seite 124)

$\mathbf{M}$	Kontaktmoment, im Sensor gemessen (Seite 162)
$\mathbf{M}$	Massenmatrix (Seite 17)
n	Zahl der Elemente (Seite 124) (Indizes auch ab Seite xvii)
n	Zahl der Unbekannten im Kalman-Filter (Seite 271)
n_t	Totzeit in Abtastschritten (Seite 131)
N	bei prädiktiver Regelung: Prädiktionshorizont in Abtastschritten (Seite 20)
N	Zahl der Abtastschritte (Seite 55)
N	Zahl der Messgleichungen im Kalman-Filter (Seite 274)
$\mathbf{p}$	Vektor der Merkmalspositionen im Raum der Bildkoordinaten(Seite 26)
p_{0i}	Element der Diagonalmatrix $\mathbf{P}_0$ (Seite 273)
$\mathbf{P}$	Kovarianzmatrix des Schätzfehlers beim Kalman-Filter (Seite 273)
$\mathbf{P}_0$	Anfangswert der Kovarianzmatrix (Seite 273)
$\mathbf{P}^*$	Kovarianzmatrix nach der Prädiktion (Seite 274)
$\mathbf{q}$	Vektor der Gelenkwinkel (Indizes ab Seite xvii) (Seite 17)
q	(gleicher) Wert der Elemente der Diagonalmatrix $\mathbf{Q}$ (Seite 286)
$\mathbf{Q}$	Kovarianzmatrix der Änderungen der Schätzwerte beim Kalman-Filter (Seite 272)
r	Reglerparameter (Seite 155) (Indizes ab Seite xvii wie bei n, mit zusätzlichem Laufparameter, z. B. r_{wi} mit $i = 1, \cdots, n_w$)
$\mathbf{r}$	Richtungsvektor (Seite 298) (Indizes ab Seite xvii wie bei x)
$\mathbf{R}$	Rotationsmatrix (Seite 23)
$\mathbf{R}$	Verstärkungsmatrix, die u. U. auch eine Rotation enthält (Seite 24) (Indizes ab Seite xvii wie bei K)
$\mathbf{R}$	Reglerübertragungsfunktion (Seite 147)
$\mathbf{s}$	Vektor der Sensorwerte (Seite 24)

s_e	Gewichtung der Regeldifferenzen (Seite 139)
s_u	Gewichtung der Stellgrößen (Seite 139)
t	Zeitpunkt (Seite 20)
T_0	Abtastzeit (Seite 129)
$u, \mathbf{u}$	Stellgröße, Vektor der Achswerte (Seite 101) (Indizes ab Seite xvii)
Δu	Änderung der Stellgröße durch die Optimierungsstufe (Seite 136)
$\mathbf{U}_i$	Dreiecksmatrix aus der Zerlegung der Kovarianzmatrix $\mathbf{P}$ (Seite 278)
$\mathbf{v}$	Systemrauschen beim Kalman-Filter (Seite 271)
v_0	Vorschub bei Konturverfolgung (Seite 298)
$\mathbf{V}$	(vektorielle) Funktion der Coriolis- und Zentrifugalkräfte (Seite 17)
$w, \mathbf{w}$	Sollwert, Vektor der Achswerte (Seite 101) (Indizes ab Seite xvii wie bei $\mathbf{y}$)
w	Messrauschen beim Kalman-Filter (Seite 271)
$\mathbf{x}$	Zustandsvektor (Seite 168)
$\mathbf{x}$	Vektor der kartesischen Positionen (Seite 23) (Indizes ab Seite xvii wie bei $\mathbf{q}$)
$\mathbf{x}_0$	Zentrum einer Gauß'schen Glocke bei radialen Basisfunktionen (Seite 31)
x'	Eingang der Aktivierungsfunktion bei Neuronalen Netzen (MLP) (Seite 31)
$y, \mathbf{y}$	Ist-Wert, Vektor der Achswerte (Seite 101) (Indizes ab Seite xvii)
z	Operator der zeitlichen Verschiebung (Seite 34)
γ	Eingangsfunktion der Dynamikgleichung beim Kalman-Filter (Seite 271)
δ	Impulsfunktion (Seite 279)
δ	Hilfsgröße beim Training von Neuronalen Netzen (MLP) (Seite 284)

θ	Schätzvektor (Seite 271)
σ	Standardabweichung der Messstörung (bei Schätzungen) (Seite 272)
$\mathbf{\Sigma}$	Selektionsmatrix (Seite 23)
τ	Vektor der Motormomente (Seite 17) (Indizes ab Seite xvii wie bei $\mathbf{u}$)
$\mathbf{\Phi}$	Selektionsmatrix bei kartesischer Adaption von Achswerten (Seite 188)
$\mathbf{\Phi}$	Systemmatrix der Dynamikgleichung des Kalman-Filters (Seite 271)
ψ	Hilfsvektor (Messvektor) bei Schätzungen (Seite 271)
τ^c	Vektor der Achsmomente, die den Kontaktkräften entsprechen (Seite 162)
ω	Kreisfrequenz (Seite 141)

Indizes

Indizes, die bei unterschiedlichen Variablen dieselbe Bedeutung haben werden nur einmal beispielhaft aufgeführt.

$\mathbf{K}_D$	Verstärkungsfaktor für D-Anteil (Seite 18)
$\mathbf{K}_P$	Verstärkungsfaktor für P-Anteil (Seite 18)
n_a	Zahl der Parameter a_i der Streckendarstellung (Seite 158)
n_b	Zahl der Parameter b_i der Streckendarstellung (Seite 158)
n_c	Zahl der Parameter der Vorsteuerung zur Gewichtung zukünftiger Kontaktmomente (Seite 163)
n_e	Zahl der Parameter des Reglers zur Gewichtung von e (Seite 158)
n_f	Zahl der Parameter des Reglers zur Gewichtung vergangener Kontaktmomente (Seite 164)
n_g	verwendete Zahl der Parameter der Gewichtsfunktion (Seite 132)

n_{kij} Zahl der Parameter des Reglers zur Gewichtung der Kopplungen von Achse j auf Achse i (Seite 183)

n_m geschätzte Zahl der Parameter des Modells (Seite 131)

n_o Zahl der optimierten Stellgrößen (Seite 145)

n_p Zahl der Parameter des Reglers zur Prädiktion (Extrapolation) des Verlaufs der Soll-Bahn (Seite 226)

n_u Zahl der Parameter des Reglers zur Gewichtung von u (Seite 158)

n_w Zahl der Parameter der Vorsteuerung zur Gewichtung von w (Seite 89)

$\mathbf{q}$ aktueller Ist-Gelenkwinkel (bei elastischen Robotern: antriebsseitig) (Seite 17)

$\mathbf{q}_a$ aktueller abtriebsseitiger Gelenkwinkel (arm) (Seite 105)

$\mathbf{q}_c$ kommandierter Gelenkwinkel (commanded) (Seite 90)

q_{ci} kommandierter Gelenkwinkel für Achse i (Seite 150)

$q_{ci,kj}$ Anteil der Koppelterme von Achse j am kommandierten Gelenkwinkel für Achse i (Seite 183)

$q_{ci,kk}$ Anteil der Kontaktkraftkompensation am kommandierten Gelenkwinkel für Achse i (Seite 150)

$q_{ci,nn}$ Anteil des Neuronalen Netzes am kommandierten Gelenkwinkel für Achse i (Seite 150)

$q_{ci,neu}$ Sollwert beim Training des kommandierten Gelenkwinkels für Achse i (Seite 153)

$q_{ci,pv}$ Anteil der linearen Positionsvorsteuerung am kommandierten Gelenkwinkel für Achse i (Seite 150)

$\mathbf{q}_d$ Soll-Gelenkwinkel (desired) (Seite 17)

$\mathbf{u}_{neu}$ korrigierte (adaptierte) Stellgröße (Seite 101)

x_n Komponente des Vektors $\mathbf{x}$ normal zur Kontur (Seite 299)

$\mathbf{x}_p$ bei prädiktiver Regelung: prädizierte Position (Seite 20)

$\mathbf{x}_p$ vorgegebener Bahnpunkt (Seite 109)

$\mathbf{x}_r$ Referenzposition (Seite 109)

$\mathbf{x}_s$ sensorische Soll-Position (Seite 109)

$\mathbf{x}_s$ Schätzpunkt beim Tabellenspeicher ME9 (Seite 169)

x_t Komponente des Vektors $\mathbf{x}$ tangential zur Kontur (Seite 299)

$\mathbf{y}_x$ kartesischer Stellgrößenvektor (Seite 56)

y_0 Arbeitspunkt, Gleichgewichtspunkt des Ist-Wertes (Seite 128)

Kapitel 1

Einleitung

1.1 Motivation

In der industriellen Fertigung werden heute Roboter eingesetzt, um Handhabungen und Fertigungsaufgaben auszuführen, die ein Mensch nicht oder nur mit begrenzter Genauigkeit und Geschwindigkeit erledigen könnte, und für die sich der Bau von fest verdrahteten Fertigungsautomaten nicht lohnt. Dies sind z. B. Arbeiten mit hohem Kraftaufwand oder hochgenaue Bearbeitungsaufgaben, zu denen der Mensch nicht oder nur mit Hilfsmitteln imstande ist.

Weitere Anwendungsgebiete sind überall dort zu finden, wo der Einsatz von Maschinen weniger Kosten erfordert als eine menschliche Arbeitskraft. Dazu zählen einerseits Aufgaben in lebensfeindlicher Umgebung, in denen für einen Menschen aufwendige Schutzmaßnahmen getroffen werden müssten. Dies sind z. B. Tätigkeiten unter Wasser, im Weltraum oder in für den Menschen unverträglichen Temperatur-, Atmosphären- oder Strahlungsbedingungen. Andererseits lohnt sich der Einsatz von Robotern, wenn Roboter die Arbeit bei gleicher Güte in, bezogen auf die Betriebs- bzw. Lohnkosten, geringerer Zeit ausführen können. In der industriellen Fertigung liegt dieser Fall aufgrund der schnellen Bewegungsmöglichkeiten und der hohen Wiederholgenauigkeit oft vor.

Jüngste Prognosen über die Entwicklungen in der Industrierobotik (z. B. [3]) besagen, dass die Zukunft der Robotik maßgeblich durch die Weiterentwicklung der Steuerung (Regelung) und den Einsatz von hochauflösenden und echtzeitfähigen Sensoren bestimmt wird.

Kritische Einsatzfälle liegen immer dann vor, wenn ein Roboter die Aufgabe

nicht zufrieden stellend ausführen kann, sei es aufgrund einer ungenau oder falsch angenommenen Umgebung oder aufgrund von Ungenauigkeiten der Robotersteuerung. Heute noch problematische Aufgaben sind das Einsetzen von Stiften in Passungen oder das Ansetzen von Schrauben. Bei solchen Aufgaben können aufgrund der hohen Steifigkeit heutiger auf dem Markt verfügbarer Industrieroboter bei ungenauer Bewegung unerwünscht hohe Kontaktkräfte auftreten. Weitere kritische Anwendungen sind Aufgaben, bei denen der Roboter zwar nicht im Kraftkontakt zu seiner Umgebung steht, die aber trotzdem sehr genau ausgeführt werden müssen, z. B. das Schneiden oder Löten mit Hilfe eines Lasers oder das schnelle Auftragen von Kleber für eine schmale Dichtung. Bei diesen Aufgaben sind während der gesamten Ausführung der Bahn enge Toleranzen einzuhalten.

Damit sind zwei Anwendungsbereiche genannt, in denen bezüglich der Robotersteuerung Handlungsbedarf besteht. Einerseits müssen die Roboter mit sensorischen *Sinnen* ausgestattet werden, um ohne Unterstützung des Menschen auf unerwartete kritische Situationen reagieren zu können (Autonomie), und andererseits müssen die Bewegungen der Roboter dem Anwendungsprogramm und seinem Anforderungen entsprechen. Während die Verarbeitung von taktilen und nichttaktilen Sensordaten im Rahmen dieser Arbeit nur am Rande betrachtet wird, soll die Frage der Genauigkeit und Bahntreue der Trajektorien schneller Roboteranwendungen näher untersucht werden.

Dabei zeigt sich, dass ein Roboter bei Bearbeitung gleicher Aufgaben aufgrund gleicher Bewegungsbefehle zwar nahezu gleiche Bahnen ausführt, dass diese messbaren Bahnen sich aber deutlich von den durch das Anwendungsprogramm vorgegebenen Soll-Bahnen unterscheiden.

1.2 Problemstellung

Die Steuerung eines Roboters wird üblicherweise in drei Aufgabenbereiche eingeteilt (siehe auch Abb. 1.1). Die offline durchgeführte *Planung* generiert und definiert die Soll-Bewegung (Positionen, Orientierungen und Geschwindigkeiten zu verschiedenen Zeitpunkten) des Endeffektors bezüglich des zu manipulierenden Objekts und bezogen auf die als Modell vorgegebene Umwelt. Das durch die Planung vorgegebene bzw. generierte Programm wird im folgenden als gegeben angenommen. Die *Regelung* sorgt in Echtzeit für die Ausführung der Soll-

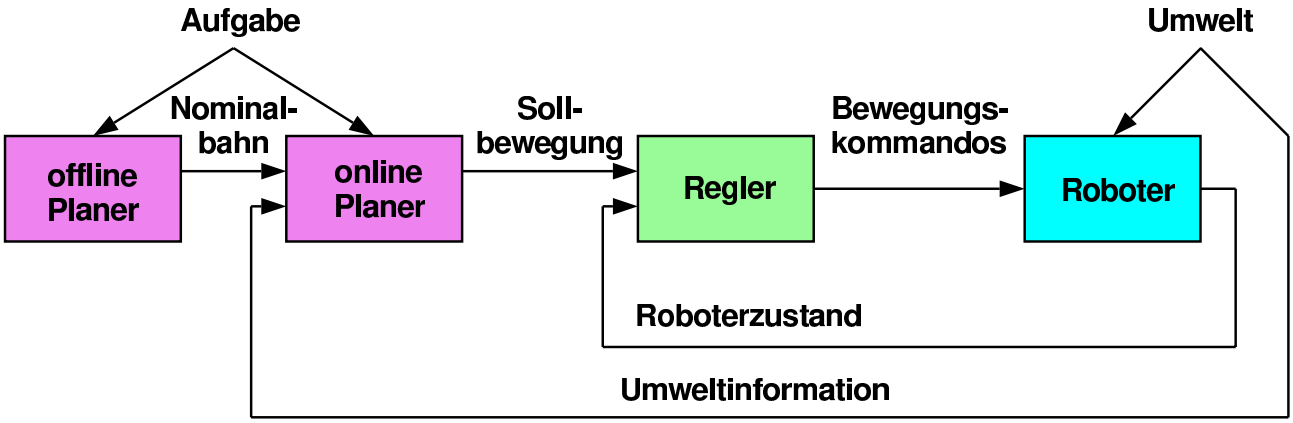


Abbildung 1.1: Struktur der Steuerung von Robotern

Bewegung und die Sicherstellung der Bahngenauigkeit innerhalb vorgegebener Toleranzen. Wenn die Abbildung der Orts- und Lagekoordinaten der zu manipulierenden Objekte auf das Roboterbezugskoordinatensystem nicht vollständig bekannt oder zeitvariant ist, wird dazwischen noch ein weiterer Funktionsblock benötigt, der kurz als online Planer bezeichnet wird. Er bewirkt eine sensorgestützte Modifikation der offline geplanten Bahn. Auf die Fragestellung der dynamischen Bahnplanung und -korrektur wird später kurz eingegangen. Schwerpunkt dieser Arbeit ist aber die Untersuchung und Entwicklung einer adaptiven Regelungsstrategie, die dafür sorgt, dass die Bewegungen des Roboters in der geplanten Weise mit einer vorgegebenen Genauigkeit ausgeführt wird.

Diese Sicht der Regelung geht über die in der Robotik üblichen Lage- und Bahnregelungsverfahren hinaus. So umfasst sie neben der Rückführung von Messwerten auch Algorithmen und Verhaltensstrategien, die unabhängig von aktuellen Roboterzuständen durchgehend aktiviert sind. Die hierzu notwendigen Algorithmen können dabei online oder offline parametriert werden. Aus diesem Grund wird das Regelungssystem in Abb. 1.2 weiter unterteilt in eine Vorsteuerung, die die Bewegungsbefehle unabhängig vom aktuellen Roboterzustand in Bezug zu der geplanten Bahn bestimmt, und in eine Regelungsschleife, in der ohne Kenntnis der Vorsteuerung, dafür aber meist mit höherer Abtastrate, der Roboterzustand dem aktuellen Soll-Zustand nachgeführt wird. Somit beschreibt der Soll-Zustand nur ein Abtastintervall der geplanten Bahn.

Sowohl die umweltabhängige Planung als auch die Vorsteuerung werden bei der überwiegenden Mehrheit heutiger Industrieroboteranwendungen durch explizite Vorgabe (Teach-in) von Bahnpunkten vorgenommen [216]. Damit werden die Bahnen als Folge von Punkten gespeichert, die entweder beliebig oder durch

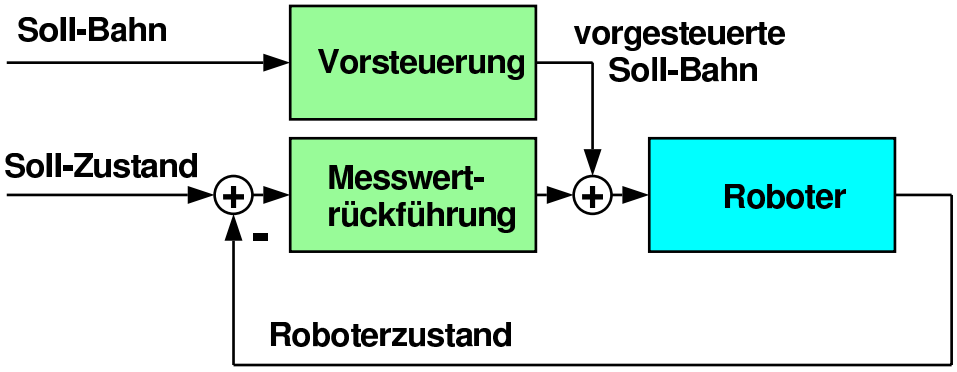


Abbildung 1.2: Darstellung des Regelkreises

Geraden verbunden werden. Für eine offline Korrektur der resultierenden Trajektorie zur Kompensation von Bahnfehlern bleibt nur die Verschiebung dieser Teachpunkte, bzw. das Einfügen von Zwischenpunkten. Dies wird in der Praxis meist manuell ausgeführt. Eine durch explizites Vormachen am Roboter erzeugte Bahn wird ausgehend vom Einzelschrittbetrieb iterativ derart korrigiert, dass sie bei immer höheren Geschwindigkeiten bis zu dem durch Hardwarerestriktionen bestimmten Maximum die vorgegebene Soll-Bewegung mit ausreichender Genauigkeit ausführt. Es wird also eine heuristisch spezifizierte Bahn eingegeben, die als Grundlage der Regelung zum Ausführen der Soll-Bahn dient.

Eine andere Möglichkeit zur Bahnplanung stellt die offline Programmierung von Roboterbewegungen mit Hilfe eines CAD-Systems [24] dar. Dazu gibt es eine Reihe von Programmiersystemen, wie z. B. Delmia/IGRIP [67], CATIA [34], KISMET [130] oder Produkte von z. B. Tecnomatix [235] (ehemals ROBCAD, AnySIM u. a.). Solche Systeme erlauben die interaktive Definition und Visualisierung von Bewegungen, berücksichtigen aber weder statische noch dynamische Bahnabweichungen. Eine offline Bahnplanung mit Hilfe von CAD-Systemen kann nach heutigem Entwicklungsstand das Teach-in an realen Robotern zur Korrektur und Optimierung des Programms nicht ersetzen. Das Teach-in kann aber mit Hilfe von Rechnern vereinfacht werden, da die Bahn meist nur geringfügig modifiziert werden muss.

Zur Vermeidung von Notationsproblemen wird für Roboter mit Schnittstellen zur Spezifikation von Bahnpunkten der Begriff einer *kommandierten* Bahn eingeführt, die Eingabewerte der Sollwerte für die Lageregler bezeichnet, während die *Soll*-Bahn die eigentlich gewünschte Bewegung beschreibt. Dies widerspricht der Notation herkömmlicher Robotersteuerungen, die keine Unterschiede zwischen Kommandos und Sollwerten machen und meist den Begriff Sollwert be-

nutzen. Beim klassischen Teach-in, bei dem ein Bediener die (kommandierten) Bahnpunkte relativ zur Umgebung festlegt, ist die Soll-Bahn meist nicht quantitativ bekannt. Bei offline mit Hilfe von CAD-Systemen geplanten Bahnen wird dagegen eine meist analytisch vorgegebene Soll-Bahn generiert, für die im Teach-in nur noch Bahnstützpunkte und die entsprechenden Bewegungsbefehle bestimmt werden müssen.

Grundsätzlich muss man zwischen statischen und dynamischen Bahnabweichungen unterscheiden. Statische Fehler treten auch bei langsamem Abfahren der vorgegebenen Bahn auf, z. B. durch elastische Verformung der Roboterarmelemente aufgrund der Schwerkraft. Diese Effekte sind bei offline geplanten Bahnen besonders störend, da bei direkter Bahnvorgabe am Roboter schon die korrigierten Punkte eingegeben werden. Statische Bahnabweichungen lassen sich i. A. durch sensorgestützte Bahnkorrektur kompensieren. Sie werden in dieser Arbeit als ungenaue Transformation auf das Roboterbezugskoordinatensystem betrachtet, deren Berücksichtigung nach Abb. 1.1 zur online[1] Bahnplanung gehört.

Daneben gibt es dynamische Bahnfehler, die durch das dynamische Verhalten der Roboteranordnung erzeugt werden. Dynamische Fehler treten unabhängig von der ursprünglichen Vorgabe der Bahnstützpunkte auf. Sie machen sich insbesondere bei sehr hohen Geschwindigkeiten bemerkbar. Eine sensorgestützte Kompensation dynamischer Bahnfehler ist meist nicht möglich, da jede Änderung der Kommandos sich aufgrund der Dynamik erst mit Verzögerung auswirkt. Dies wurde von Hirzinger in [102] als „Raum-Zeit-Problem“ bezeichnet. Im Gegensatz zur Modifikation der Bewegung aufgrund der meist langsamen Korrektur der Umweltkenntnis können die dynamischen Bahnabweichungen durch konventionelle online Rückführung der sensorisch erfassten Regelabweichung kaum reduziert werden. Stattdessen erfordern dynamische Bahnabweichungen die oben erwähnte oft aufwendige Korrektur der Positionskommandos. Diese Arbeit beschäftigt sich deshalb mit der „Vorhersage“ und, daraus resultierend, mit der prädiktiven Kompensation der dynamischen Abweichungen bei großen Verfahrgeschwindigkeiten.

Teach-in Programmierverfahren stellen bei industriellen Industrieroboteranwendungen einen wichtigen Kostenfaktor dar, da zumindest die statische oder

[1] Mit *online* Planung ist gemeint, dass im Gegensatz zur offline Programmierung das reale System benötigt wird. Bei mehrfacher Aufgabenausführung *ohne Änderung der Umwelt* sind die online Korrekturen jeweils gleich, werden also sinnvollerweise nur einmal bestimmt.

dynamische Korrektur nur am Zielsystem, also am realen Roboter vorgenommen werden kann, was den Stillstand der betreffenden Produktionsanlage bedingt. Der zeitliche Aufwand bei Teach-in basierten Korrekturmaßnahmen und damit die Dauer des Produktionsausfalls ist von den Genauigkeitsanforderungen abhängig, wodurch der Einsatz von Robotern für Aufgaben, die präzise ausgeführt werden müssen, bislang noch sehr beschränkt ist.

Als Alternative zu hochgenauen Teach-in Korrekturverfahren wurden Ende der Siebzigerjahre passive Nachgiebigkeiten für Fügeaufgaben vorgeschlagen [257]. Dadurch lassen sich Montageaufgaben ohne adaptive online Planung trotz geringer Genauigkeit des Roboters aber durch gezielt eingesetzte mechanische Nachgiebigkeiten ausführen. Ihr Anwendungsbereich ist aber beschränkt, sofern keine unterstützende Sensorik eingesetzt wird. Bei der Sensordatenverarbeitung ist dagegen die erzielbare Bahngeschwindigkeit normalerweise durch die Abtastrate bzw. Zykluszeit der Robotersteuerung und die Bandbreite der Sensorik begrenzt.

In jüngerer Zeit wurden bei Robotersteuerungen die Möglichkeiten zur Eingabe kontinuierlicher Bewegungen verbessert. Große Beachtung wurde dabei dem Verhalten des Roboters an den programmierten Punkten gewidmet („überschleifen“ zwischen den Geradenstücken). Aber auch damit konnte noch keine befriedigende Bahngenauigkeit garantiert werden.

Die bei dem DLR in Oberpfaffenhofen verwendete Repräsentation von Soll-Bahnen als Folge von Abtastpunkten erlaubt dagegen eine flexiblere Korrektur. Als Beispiel sei die Steuerung eines Roboters durch die so genannte Space Mouse [102, 101] genannt, die, ohne offline Programmierung, dem Benutzer eine direkte Eingabe von Bewegungen für einen Roboter ermöglicht. Dabei erweist es sich als hilfreich, wenn Kräfte auf den Bediener haptisch zurückgekoppelt werden, wie z. B. bei dem Master-Slave System PHANToM [168, 222], bei kraftreflektierenden Joysticks [5] oder bei dem Teach-device [131] des DLR (Abb. 1.3 bis Abb. 1.6).

Aber auch bei flexibler Bahnvorgabe wie der Folgeprogrammierung treten bei der Ausführung Bahnfehler auf, wenn der Roboter sich bei der späteren Ausführung schneller als bei dem Teach-in Prozess bewegen soll. Im Falle der Verfügbarkeit eines exakten Robotermodells kann für eine quasi kontinuierliche Trajektorie eine geeignete Stellgröße berechnet werden, die die gewünschte Soll-Trajektorie genau reproduziert. Dieser als inverses dynamisches Problem bezeichnete Ansatz setzt einerseits eine genaue Modellkenntnis voraus und führt

Abbildung 1.3: Space Mouse

Abbildung 1.4: Kraftreflektierender Joystick des DLR

andererseits zu stark verkoppelten nichtlinearen Gleichungen [10, 221]. Damit wäre die Ausführung einer offline geplanten Bahn ohne weiteres Teach-in möglich. In der Regel liegt aber kein hinreichend exaktes Robotermodell vor.

Einerseits gibt es statische Unsicherheiten, z. B. aufgrund von ungenauer Vermessung der Roboterkinematik oder der Gelenkelastizitäten, andererseits gibt es zeitvariante Größen, wie die Reibung, die sich nicht nur schwer bestimmen lassen, sondern sich auch noch während des Betriebs ändern können. Aus diesem Grund ist es grundsätzlich sinnvoll, das Modell und den Regler des Roboters kontinuierlich zu adaptieren. Dies erfordert aber im Gegensatz zur reinen offline Programmierung auch die Einbeziehung des realen Zielsystems. Bei der Einbeziehung des Zielsystems sowie der Verfügbarkeit einer adaptiven Steuerung ist

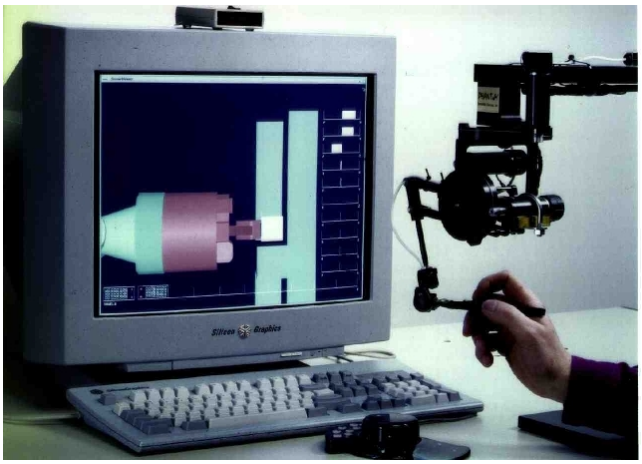


Abbildung 1.5: PHANToM Eingabegerät mit Kraftrückkopplung

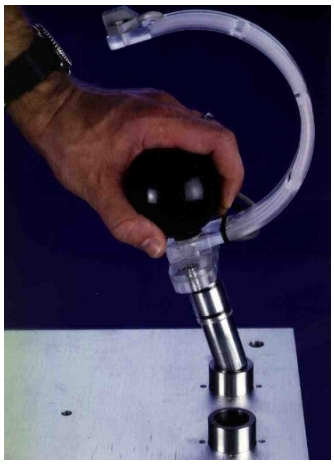


Abbildung 1.6: Teach-device des DLR

ein effektives Teach-in denkbar, das kurze Programmierzeiten bei hoher Bahntreue garantieren kann.

Dies gilt für das in dieser Arbeit vorgeschlagene *bahnunabhängige* Verfahren, bei dem die erforderliche Information über das Zielsystem bei der Installation bzw. den üblichen Wartungsarbeiten gewonnen werden kann, nicht aber bei der Umstellung auf eine neue Aufgabe. Hierzu ist es sinnvoll, die interaktive Korrektur der Stütz- und Abtastpunkte zu automatisieren und ein Gütemaß oder Gütefunktional zu definieren, das eine selbständige Adaption und Optimierung steuern kann.

1.3 Zielsetzung und Beitrag der Arbeit

Ziel der Arbeit ist es,

- die übliche Art der durch aufwendige manuelle Korrektur der Bahnstützpunkte charakterisierte Programmierung einer Roboterbewegung zu vereinfachen,
- die Ausführung der programmierten Bahnen zu optimieren und damit eine höhere Bahngenauigkeit zu garantieren als dies bei der heute noch weit verbreiteten Art des Teach-in Programmierens möglich ist,
- roboterunabhängig, offline programmierte Bewegungen adaptiv an das Zielsystem anzupassen.

Dadurch sollen die Anwendungsbereiche von Industrierobotern um die diejenigen Aufgaben erweitert werden,

- deren Programmierung bislang bezogen auf den zeitlichen Optimierungsaufwand zu aufwendig war,
- die bisher aus Genauigkeitsgründen nur langsam (unwirtschaftlich) ausgeführt werden konnten,
- bei denen zeitvariante Umwelteinflüsse wie die Temperatur, Alterungserscheinungen oder sonstige schwer erfassbare, die Genauigkeit bestimmende Parameter zur Überschreitung der erlaubten Toleranzen führten.

Der Beitrag der Arbeit ist also die Entwicklung eines adaptiven Systems, das die Bahngenauigkeit eines Industrieroboters automatisch erhöht. Dazu wird zunächst eine hierachische Steuerungssystemarchitektur vorgeschlagen, die mehrere überlagerte Regelkreise integriert. Der äußere Regelkreis korrigiert durch Verarbeitung sensorischer Informationen die statischen Ungenauigkeiten der gewünschten Roboterbewegung. Im inneren Teil der Regelungshierarchie werden vorgegebene Trajektorien betrachtet und adaptiv angepasst. Dabei kann die von Roboterherstellern üblicherweise mitgelieferte Steuerung integriert werden. Dabei wird nicht nur die vorhandene Hardware genutzt, sondern das adaptive System kann in die vorgegebenen Softwarestrukturen eingebunden werden.

In Erweiterung zu den in der Literatur zum Erlernen von Trajektorien (siehe Abschnitt 2.4.5) vorgestellten Verfahren werden in dieser Arbeit nicht nur individuelle Trajektorien verbessert, sondern ein allgemeingültiger Ansatz wird verfolgt. Darüber hinaus wird das gesamte Verhalten des Roboters bei schnellen Bewegungen erfasst. Nach Planung einer neuen Bahn können die für diese erforderlichen Bewegungsbefehle ohne weitere Adaption durch einfache Rechnung bestimmt werden[2]. Damit ist das vorgeschlagene Verfahren auch bei Anwendungen interessant, bei denen keine vielfachen Wiederholungen der Bewegung vorkommen, also bei Kleinserienfertigungen, bei denen Roboter aufgrund von zu langen Umrüstzeiten bisher unwirtschaftlich waren. Es ist sogar möglich, bei online geplanten oder modifizierten Bewegungen die dynamischen Effekte zu kompensieren und den Roboter ohne Teach-in einzusetzen.

Weiterhin wird durch die vorgeschlagene Optimierung der Einsatz von Robotern für Aufgaben ermöglicht, bei denen die erforderliche Genauigkeit bisher ohne besondere Maßnahmen nicht garantiert werden konnte. Kapitel 3 zeigt, dass die Genauigkeitsabweichungen bei Ansteuerung von Robotern mit der dazugehörigen industriellen Steuerung bei hoher Geschwindigkeit einige Millimeter betragen können. Der Beitrag dieser Arbeit ist die weitgehende Kompensation dieser Bahnabweichungen.

Das gilt auch für Robotereinsätze im Dauerbetrieb, bei denen die Abnutzung oder Umwelteinflüsse das Roboterverhalten verändern und damit eine Adaption der Regler erfordern. Hier liegt auch der Vorteil gegenüber modellgestützten Verfahren. Selbst wenn ein genaues analytisches Modell vorliegt, so gilt es nur

[2]Von manchen Autoren, z. B. Tolle [238] wird ein adaptives System, das nach einer Anpassungsphase keine weitere Adaption benötigt, als lerndendes System bezeichnet. Nach dieser Notation müssen adaptive Regler von nichtlinearen Systemen bei Veränderung des Arbeitspunktes wieder neu adaptiert werden. In dieser Arbeit wird die Definition des adaptiven Systems nicht so eng gefasst.

für den Nominalfall, nicht aber für die aktuellen Parameter des verwendeten individuellen Roboters.

Bei den bisher weit verbreiteten Verfahren der Teach-in basierten Roboterprogrammierung wird die Bewegung des Roboters manuell oder automatisch derart modifiziert, dass die Bahnfehler während der Testläufe reduziert werden. Treten jedoch andere Störungen auf, so verursachen diese i. A. zusätzliche Bahnabweichungen. Dies erweist sich insbesondere bei Bearbeitungsaufgaben als hinderlich, da dadurch z. B. beim Entgraten von Gussteilen die wirkliche Bewegung des Roboters von der Stärke des Grates abhängt. Die Unabhängigkeit der Bewegung von (sensorisch erfassbaren) Störungen ist also ein weiterer Beitrag dieser Arbeit.

Das vorgeschlagene adaptive System wird an einigen unterschiedlichen Robotertypen entwickelt und qualitativ wie quantitativ evaluiert. Es wird gezeigt, dass es auch auf andere Roboterkonfigurationen durch seine Verallgemeinerbarkeit übertragen werden kann.

1.4 Gliederung der Arbeit

Nach dieser Einführung folgt in Kapitel 2 eine Analyse und Bewertung der wichtigsten in der Literatur vorgestellten Ansätze zur Regelung. Dabei werden neben industriell eingesetzen Reglerstrukturen auch sich noch im Entwicklungsstadium befindende Verfahren beschrieben. Dazu gehört auch ein Überblick über mögliche Adaptionsverfahren.

In der Literatur zur Lösung vergleichbarer Aufgabenstellungen konnte keine ausführliche Analyse vorhandener Industrierobotersysteme gefunden werden. In den meisten Beschreibungen von industriellen Robotersteuerungen fehlen wesentliche Angaben zur Beurteilung deren dynamischer Genauigkeit. Daher werden in Kapitel 3 zwei typische industrielle Robotersysteme auf ihre Bahngenauigkeit unter verschiedenen Bedingungen untersucht.

Aus diesen beiden Elementen, den existierenden Methoden aus Kapitel 2 und den Ergebnissen der Experimente aus Kapitel 3, ergibt sich im Kapitel 4 der Entwurf einer adaptiv vorausplanenden Steuerung (siehe Abb. 1.7). Dabei wird das entwickelte Konzept in den Rahmen eines sensorgestützt hierarchischen Systems integriert.

Die adaptiv vorausplanende Steuerung wird in Kapitel 5 näher ausgeführt. Dabei werden die möglichen Alternativen für die einzelnen Funktionsblöcke der adaptiven Steuerug diskutiert. Die Implementierung der ausgewählten Lösung wird ausführlich erläutert.

In Kapitel 6 folgt schließlich eine Bewertung aufgrund von Experimenten, die mit den Versuchen in Kapitel 3 direkt vergleichbar sind. Damit sind diese Experimente das zweite Standbein der Arbeit, da nur durch eine quantitative Untersuchung das Verbesserungspotential verifiziert werden kann.

Kapitel 7 fasst die Ergebnisse zusammen und beurteilt die erreichten Verbesserungen und die noch verbleibenden Probleme.

Im Anhang werden schließlich Methoden beschrieben, die in dieser Arbeit angewendet werden und deren Kenntnis, zumindest in der implementierten Fassung, nicht vorausgesetzt werden kann. Anhang C enthält die untersuchten Testbewegungen.

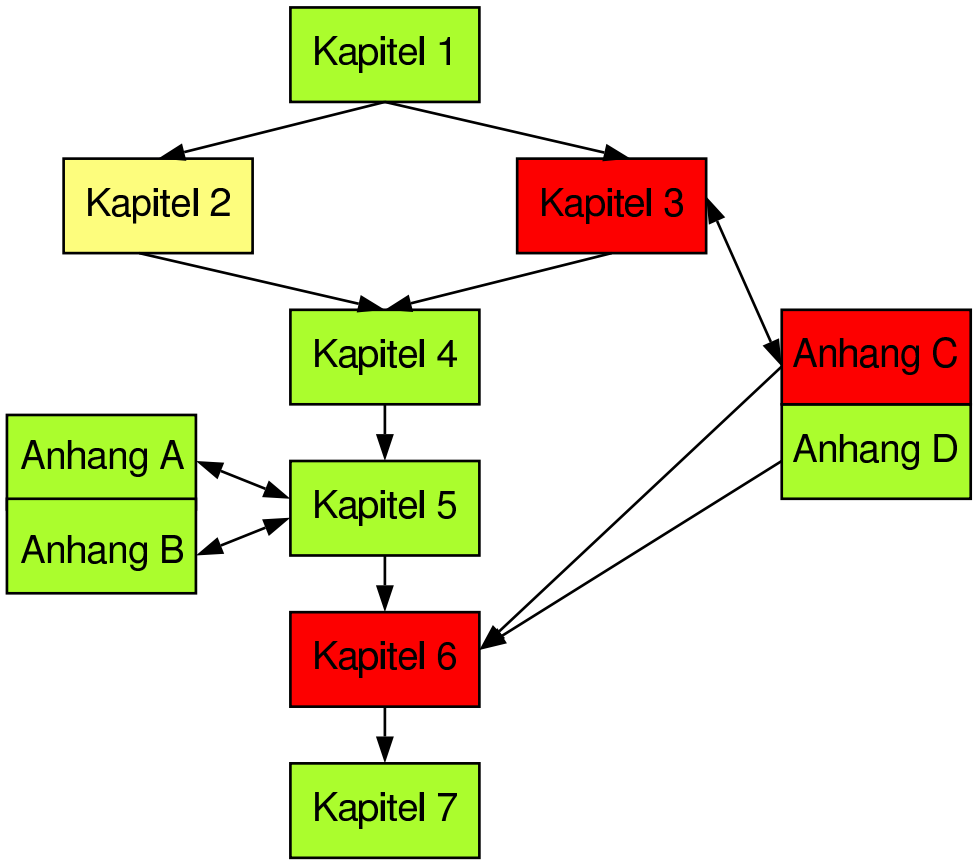


Abbildung 1.7: Aufbau der Arbeit (gelb: Literaturüberblick; rot: eigene Experimente; grün: eigene konzeptionelle Beiträge)

Kapitel 2

Ansätze zu Regelungsverfahren für bahntreue Roboterbewegungen

Dieses Kapitel beschreibt im Überblick die Ansätze, die in der Literatur zur Roboterregelung zu finden sind. Daneben wird in einer weiteren Zusammenstellung ein Überblick über die wichtigsten Adaptionsmethoden gegeben (Abschnitt 2.4).

Die meisten in der Literatur diskutierten Regelungsansätze lassen sich bei Robotern mit herkömmlicher Steuerung wegen der vorgegebenen Regelungsstruktur nicht realisieren. Deshalb wird zunächst auf Standardregelungsverfahren eingegangen, die sich schon heute mit vertretbarem Aufwand realisieren lassen, bevor der Stand der Forschung dargestellt wird. Das in dieser Arbeit vorgeschlagene Verfahren zur Verbesserung der Bahntreue geht von einer industriell verfügbaren Robotersteuerung als Basis aus.

2.1 Konventionelle Positionsregelung

2.1.1 Allgemeine Struktur

Die grundlegende Struktur von Robotersteuerungen ist in Abb. 2.1 dargestellt. Sie lässt sich durch folgende Schritte darstellen:

1. *Auswahl eines Roboterprogramms*

   In diesem ersten Verarbeitungsschritt wird der Bahnspeicher mit den kartesischen Koordinaten und Orientierungen des Endeffektors für die beim

Teach-in eingegebenen Punkte (Stützpunkte) geladen.

Für jeden dieser Punkte erfolgt dann eine

2. *Rückwärtstransformation*

   Dadurch werden die Achsstellungen, auch Roboterkoordinaten (Gelenkwinkel) bestimmt, die den Punkt und seine Orientierung repräsentieren. Aufgrund der mit meist mehrdeutigen Kinematikstrukturen behafteten Roboterkonfigurationen hängt die berechnete Achsstellung in der Regel von den benachbarten Stützpunkten der Robotertrajektorie ab, um Unstetigkeiten in der Bewegung zu vermeiden.

   Online erfolgt dann zu jedem Punkt die

3. *Interpolation des Bahnverlaufs zwischen den Bahnstützpunkten*

   Dadurch werden für jede Achse und jeden Abtastzeitpunkt der Regelung die Soll-Positionen (Soll-Gelenkwinkel) in Roboterkoordinaten festgelegt.

   Die Regelung erfolgt in der Regel als

4. *Kaskadenregelung*

   Damit sind drei ineinander verschachtelte Regelkreise für Motorstrom, Soll-Geschwindigkeit und Soll-Position gemeint. Die Differenz aus Soll-Position und aktueller Achsposition ergibt also die Soll-Geschwindigkeit, die Differenz aus Soll-Geschwindigkeit und aktueller Achsgeschwindigkeit ergibt den Sollwert für den Strom, der im innersten Regelkreis geregelt wird.

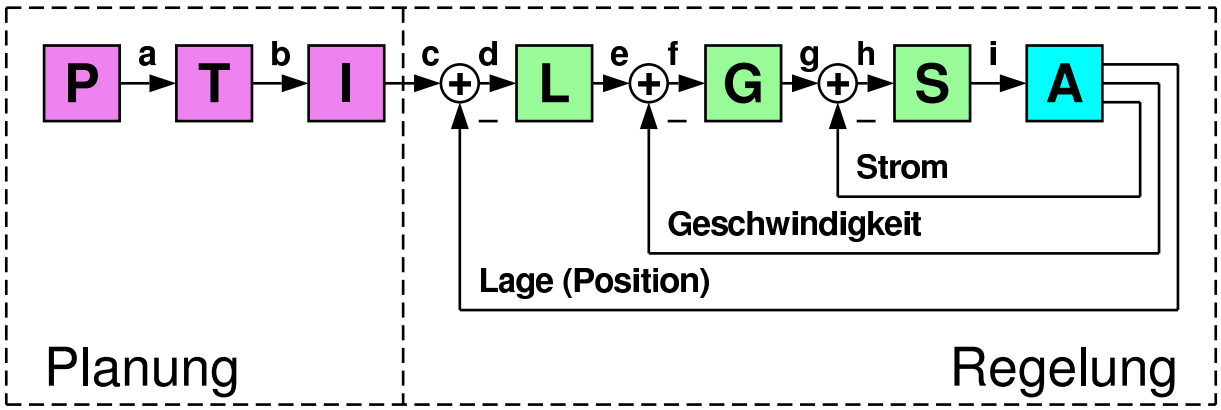


Abbildung 2.1: Struktur herkömmlicher Steuerungen ohne Sensorschnittstelle (P = Programmspeicher, T = Rückwärtstransformation, I = Interpolation, L = Lageregler, G = Geschwindigkeitsregler, S = Stromregler, A = Achse, a = Bahnstützpunkte in kartesischen Koordinaten, b = Bahnstützpunkte in Roboterkoordinaten, c = Soll-Lage im Takt der Regelung, d = Lagedifferenz, e = Soll-Geschwindigkeit, f = Geschwindigkeitsdifferenz, g = Soll-Strom, h = Stromdifferenz, i = Roboterstellgröße (Strom))

Die Bahninterpolation kann auch offline erfolgen. Dies wird aber i. A. wegen der hohen Datenmenge bei langen Bahnsequenzen und schnellem Arbeitstakt der Regelung vermieden.

Die sequentielle Reihenfolge der Funktionsblöcke *Koordinatentransformation* und *Bahninterpolation* wird oft aufgrund des hohen Rechenaufwands für die Transformation gewählt. Dadurch kann zwischen zwei Punkten des Programmspeichers im Achsraum (Gelenkraum) interpoliert werden, was bei Knickarmrobotern allerdings zu einer im kartesischen Raum gekrümmten Bahn führt.

Zur Abhilfe wird in einigen Steuerungen zweistufig interpoliert. Dabei werden im Interpolationstakt (ipo-Takt), z. B. alle 24 ms, kartesische Abtastpunkte bestimmt und in Achskoordinaten transformiert. Im Feininterpolationstakt (fipo-Takt), z. B. alle 8 ms, werden dann weitere Zwischenpunkte im Achsraum generiert. Dadurch lassen sich Abweichungen von einer kartesisch geraden Verbindung zwischen den Bahnstützpunkten des Programmspeichers unabhängig von der Länge des Bahnsegments begrenzen [137, 225].

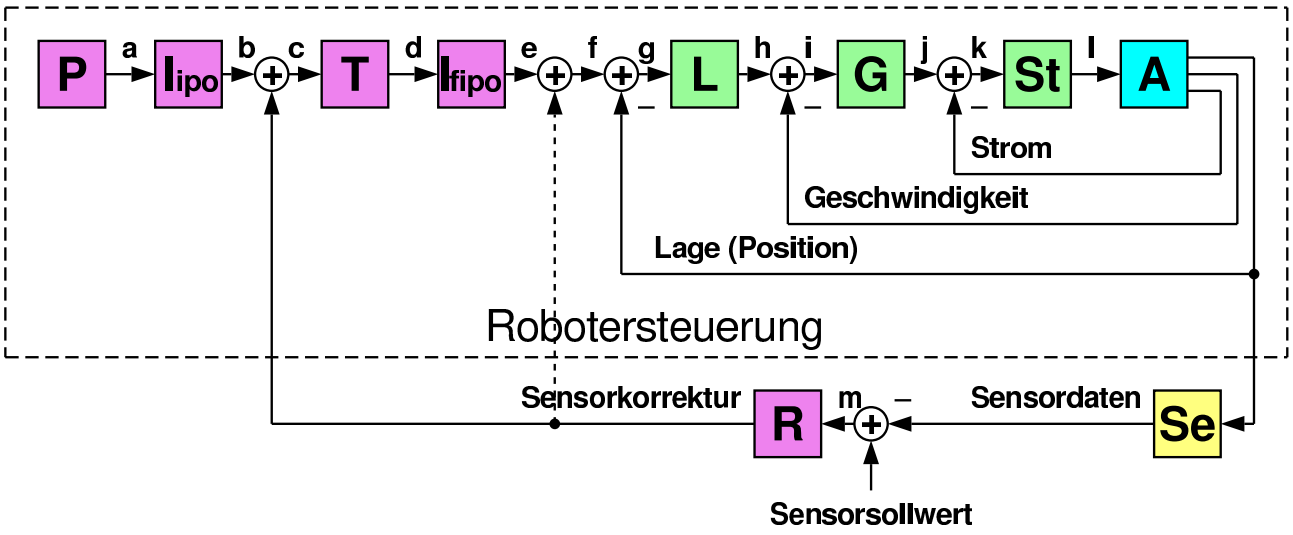


Abbildung 2.2: Struktur einer Steuerung mit Eingriffsmöglichkeit aufgrund von Sensorsignalen (P = Programmspeicher, T = Rückwärtstransformation, I = Interpolation, L = Lageregler, G = Geschwindigkeitsregler, St = Stromregler, A = Achse, Se = Sensor, R = Sensorregler, a = Bahnstützpunkte in kartesischen Koordinaten, b = kartesische Nominallage im Interpolations-Takt (Ipo), c = kartesische Soll-Lage (bei kartesischer Sensorkorrektur) im Ipo-Takt, d = Soll-Lage in Roboterkoordinaten, e = Soll-Lage im Feininterpolationstakt (fipo) der Regelung, f = Soll-Lage unter Berücksichtigung von Sensorkorrekturen in Roboterkoordinaten, g = Lagedifferenz, h = Soll-Geschwindigkeit, i = Geschwindigkeitsdifferenz, j = Soll-Strom, k = Stromdifferenz, l = Roboterstellgröße (Strom), m = Sensorregeldifferenz)

Bei der Koordinatentransformation im Interpolationstakt gibt es die Möglichkeit einer online Korrektur der Bahn aufgrund von Sensorsignalen. Dies ist in Abb. 2.2 dargestellt. Bei dieser Struktur ist anstelle der kartesischen Sensorschnittstelle auch ein Eingriff im fipo-Takt (gestrichelte Linie in Abb. 2.2) möglich, der eine geringere Zeitverzögerung im Sensorregelkreis bewirkt. Dies setzt allerdings voraus, dass die Sensorsignale außerhalb der Steuerung im fipo-Takt in Achskoordinaten transformiert werden.

Die Strukturen nach Abb. 2.1 und 2.2 haben Auswirkungen auf die Regelung. So kennt die Kaskadenregelung von der gesamten Bewegung nur den jeweils nächsten Abtastpunkt im fipo-Takt, hat also keinerlei Information über Beschleunigungen zu späteren Zeitpunkten. Weiterhin erfolgt die Kaskadenregelung für jede Achse einzeln. Dadurch ist keine Berücksichtigung von Koppeltermen wie Zentrifugal- oder Corioliskräften möglich, bei denen aufgrund einer Bewegung von Achse i eine Kraft auf Achse j ausgeübt wird [221]. Diese beiden Eigenschaften herkömmlicher Steuerungen bewirken, dass eine Regelung mit dieser vorgestellten Struktur bei schnellen Bewegungen nicht ideal sein kann. Es kommt also zu Abweichungen von der Soll-Bahn. Das gilt insbesondere bei Korrekturen über die Sensorschnittstelle, die sich i. A. erst mit einigen Schritten Totzeit auf die Regelung auswirken [137, 225].

2.1.2 Kaskadenregelung mit Vorsteuerung

Das Problem der unbekannten zukünftigen Bewegung wird abgeschwächt, wenn zum aktuellen Zeitpunkt anstelle der zukünftigen Abtastwerte der betrachteten Achse die aktuelle Soll-Geschwindigkeit und Soll-Beschleunigung bekannt sind. Damit lässt sich eine Vorsteuerung nach Abb. 2.3 aufbauen. Sie berücksichtigt, wie die bisher beschriebenen Strukturen, keine Kopplungen zwischen den Achsen [203].

Problematisch bei dieser Art der Vorsteuerung ist die Einstellung geeigneter Verstärkungen, durch die der Roboter auch schnellen Sollwertänderungen verzögerungsfrei folgt, ohne an Umkehrpunkten überzuschwingen oder allgemein Strukturschwingungen anzuregen. Daher ist die verbreitet vorgesehene Geschwindigkeitsvorsteuerung bei der Standardkonfiguration i. A. abgeschaltet [137, 225]. Eine Beschleunigungsvorsteuerung ist in der Regel nicht vorgesehen.

Zur Schwingungsdämpfung bei der Geschwindigkeitsvorsteuerung schlagen

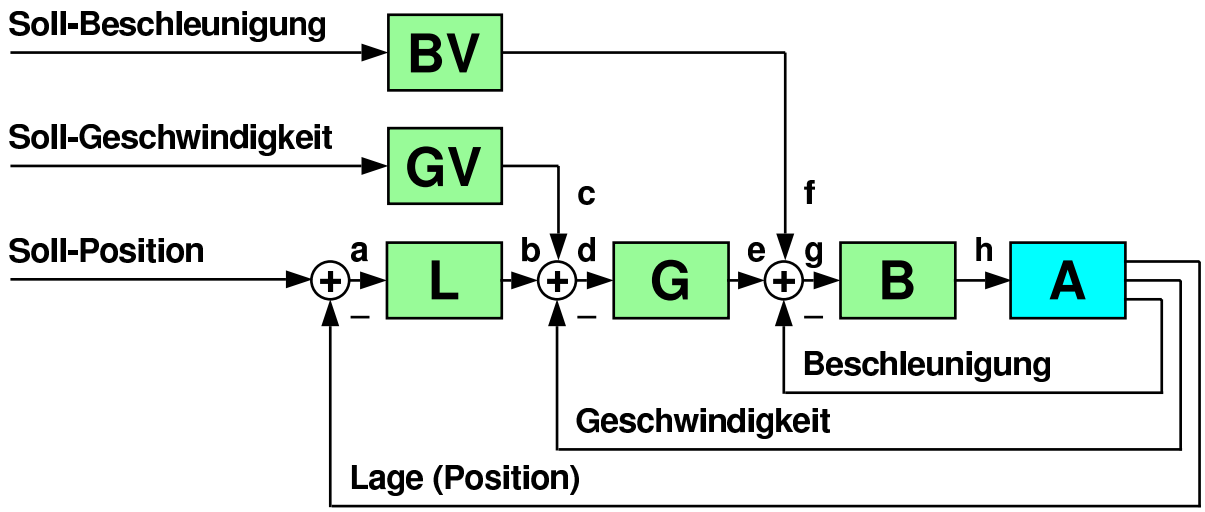


Abbildung 2.3: Dezentrale Regelung mit Geschwindigkeits- und Beschleunigungsvorsteuerung (L = Lageregler, G = Geschwindigkeitsregler, B = Beschleunigungsregler, GV = Geschwindigkeitsvorsteuerung, BV = Beschleunigungsvorsteuerung, A = Achse, a = Lagedifferenz, b = Soll-Geschwindigkeit, c = vorgesteuerte Geschwindigkeit, d = Geschwindigkeitsdifferenz, e = Soll-Beschleunigung, f = vorgesteuerte Beschleunigung, g = Beschleunigungsdifferenz, h = Roboterstellgröße (Beschleunigung))

Wörn et al. [261] die Prädiktion der Position aufgrund eines groben linearen Modells des Geschwindigkeitsregelkreises vor. Die Eingangsgröße dieses Modells ist die gewünschte Soll-Bewegung, der Ausgang ist die erwartete Ist-Bewegung. Ein Lageregler verarbeitet nur die unvorhergesehenen Positionsfehler also die Differenz zwischen beobachteten Ist-Werten und dem Modellausgang. Dadurch ergibt sich eine Entkopplung des durch die Geschwindigkeitsvorsteuerung definierten Führungsverhaltens und des durch den Lageregler bestimmten Störungsverhaltens.

2.2 Andere Verfahren der Positionsregelung

Während die bereits skizzierten Steuerungen in ihrer Grundform bereits industriell eingesetzt werden, befinden sich die folgenden Regelungsverfahren noch im Laborstadium bzw. werden als Teil industrieller Steuerungen erprobt. Hauptgrund dafür ist ihre aufwendige Reglerstruktur bei gleichzeitig hohen Abtastraten. Noch aufwendiger werden die Regelungsverfahren, wenn eine kontinuierliche Adaptionsfähigkeit vorgesehen ist (siehe Abschnitt 2.4).

2.2.1 Modellbasierte Regelung

Modellbasierte Steuerungen ersetzen die voneinander unabhängigen Kaskadenregelungen jeder einzelnen Achse durch einen komplexen Regler, der einerseits direkt die Motormomente regelt und andererseits die Kopplungen zwischen den Achsen kompensiert. Bei einem Roboter mit m Achsen wird die Soll-Bahn in Roboterkoordinaten in jedem Abtastzeitpunkt durch einen $(m \times 1)$ Lagevektor $\mathbf{q}_d$, einen $(m \times 1)$ Geschwindigkeitsvektor $\dot{\mathbf{q}}_d$ und einen $(m \times 1)$ Beschleunigungsvektor $\ddot{\mathbf{q}}_d$ berücksichtigt, wodurch theoretisch, d. h. ohne Momentenbegrenzung, eine ideale Bahnregelung möglich ist.

Sofern keine unmittelbare individuelle Momentenregelung wie bei dem im DLR entwickelten Leichtbauroboter [103, 213] möglich ist, wird angenommen, dass das ausgeübte $(m \times 1)$ Moment τ proportional zu dem Motorstrom der jeweiligen Antriebsachse ist. Die Bewegung der Roboterachsen $\mathbf{q}$ ergibt sich durch

$$\tau = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}_d + \mathbf{V}(\mathbf{q}\,\dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) + \mathbf{J}^T(\mathbf{q}) \cdot \mathbf{F}. \tag{2.1}$$

Dabei ist $\mathbf{M}(\mathbf{q})$ die $(m \times m)$ Massenmatrix, während die $(m \times 1)$ Funktionen $\mathbf{V}$ und $\mathbf{G}$ die Coriolis- und Zentrifugalterme bzw. die Gravitationseinflüsse beschreiben. Kontaktkräfte sind durch den i. A. (6×1) Vektor $\mathbf{F}$ erfasst. Sie beeinflussen die Achsen in Abhängigkeit der transponierten $(6 \times m)$ Jacobi-Matrix $\mathbf{J}$.

Theoretisch kann Gleichung (2.1) genutzt werden, um die Gelenkmomente $\tau(t)$ a priori so festzulegen, dass die gewünschten Gelenkwinkel $\mathbf{q}(t)$ abgefahren werden. Praktisch scheitert dies an der durch die zweifache Integration der Beschleunigungen instabilen Struktur. Es ist daher nötig, die Gelenkwinkel und / oder deren Ableitungen durch eine Regelung zurückzuführen. Dies ist in Abb. 2.4 ausgeführt (siehe auch [41]).

Die Regelgüte der Anordnung ist durch die ungenaue Kenntnis der in Gleichung (2.1) implizit enthaltenen Roboterparameter begrenzt. Insbesondere Reibungsgrößen sind a priori nicht bekannt und können sich darüber hinaus während des Betriebs ändern [95, 204]. Deshalb wird bei Kwon und Chung [142] die Differenz zwischen dem tatsächlichen Verhalten und dem eines einfachen Modells als Störung geschätzt und prädiktiv kompensiert.

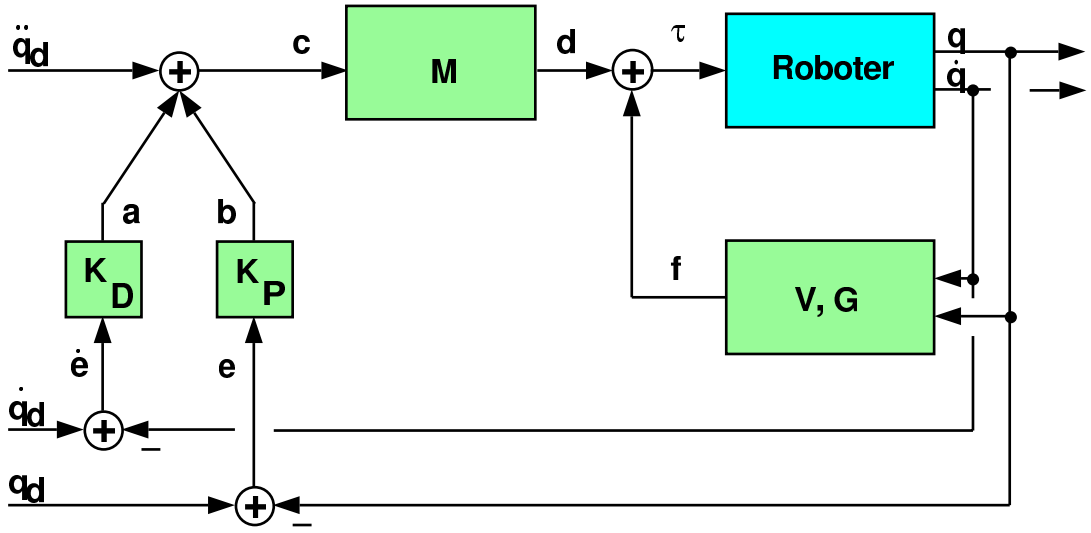


Abbildung 2.4: Modellbasierte Regelung [221] ($\mathbf{K}_P = (m \times m)$ Verstärkungsfaktor für P-Anteil des PD-Reglers, $\mathbf{K}_D = (m \times m)$ Verstärkungsfaktor für D-Anteil des PD-Reglers, a = Ausgang des D-Anteils des PD-Reglers, b = Ausgang des P-Anteils des PD-Reglers, c = aufgrund der aktuellen Regeldifferenz gewünschte Beschleunigung, d = Anteil des Motormoments zur Beschleunigung, e = Positionsregeldifferenz, f = Anteil des Motormoments zur Kompensation des Einflusses von Gravitation, Coriolis- und Zentrifugalkräften, die übrigen Bezeichnungen entsprechen Gleichung (2.1))

Bei online Bahnplanung auf der Basis von kartesisch messenden Sensorsystemen kann man einen PD-Regler auch für Regelungen in kartesischen Koordinaten einsetzen (siehe [127]). Dies beeinflusst aber die erforderliche Modellgenauigkeit nicht.

Eine weiteres Problem ist die aufgrund des zweifach integrierenden Dynamikverhaltens des Roboters erforderliche hohe Bandbreite, die oft nur eine genäherte Berechnung von Gleichung (2.1) erlaubt. Daher gibt es auch Ansätze, die die inverse Dynamik nicht im Regelkreis, sondern nur für die Soll-Trajektorie berechnen, was offline geschehen kann. Diese Struktur wird Computed Torque Regelung [10] genannt. Sie ist in Abb. 2.5 dargestellt.

Darüber hinaus führen Dresselhaus und Held in [73] die Parameter der dezentralen Regler arbeitspunktabhängig nach. Dadurch ist das Verhalten in der Praxis gleichwertig zu der Struktur mit vollständigem Modell nach Abb. 2.4.

Ein anderer Ansatz zur vereinfachten Berechnung unter Berücksichtigung aller nichtlinearen Effekte (Kopplungen) wird von Müller [184] vorgestellt und an einem Zwei-Achsen-Roboter demonstriert. Dabei werden zwar dezentrale Regler für jede Achse verwendet, es erfolgt aber auch eine Störgrößenaufschaltung zur

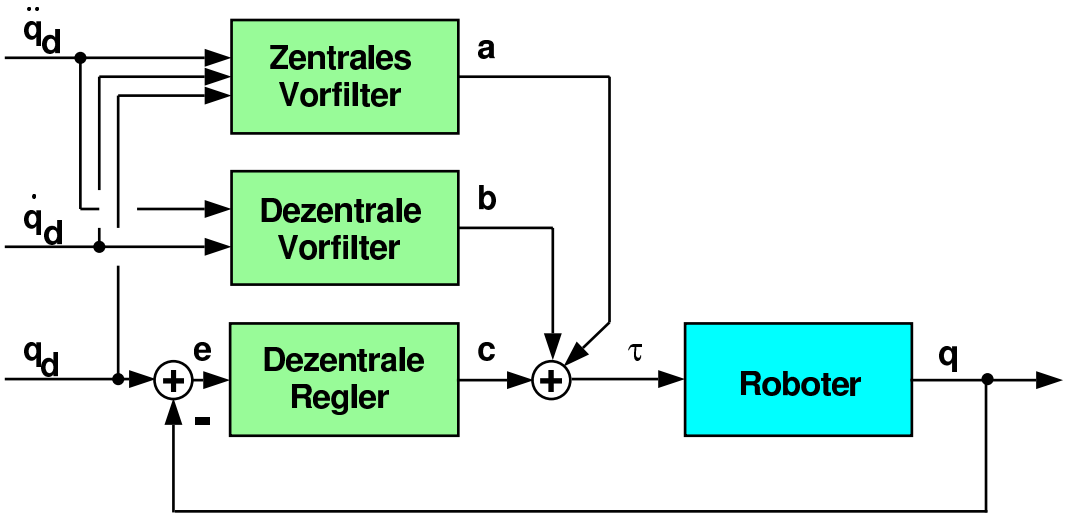


Abbildung 2.5: Computed Torque Struktur zur Vorsteuerung [221] (a = Anteil des Motormoments zur Kompensation von Kopplungen und zur Beschleunigungsvorsteuerung, b = Anteil des Motormoments zur Geschwindigkeitsvorsteuerung, c = Anteil des Motormoments zum Ausgleich der Regeldifferenz, e = Positionsregeldifferenz, die übrigen Bezeichnungen entsprechen Gleichung (2.1))

Kompensation der nichtlinearen Einflüsse. Diese Einflüsse können aufgrund der Messungen beobachtet werden.

2.2.2 Prädiktive Regelung

Bei den Methoden nach Abschnitt 2.2.1 wird die zukünftig gewünschte Bewegung nur durch die aktuelle Geschwindigkeit und Beschleunigung berücksichtigt. Bei Änderungen der Soll-Beschleunigung führt dies sehr schnell zum Erreichen der Stellgrößenbegrenzungen (Beschleunigungs- bzw. Momentengrenzwerte). Abhilfe schafft eine explizite Betrachtung einiger zukünftiger Abtastschritte der Soll-Bahn, wie dies in der prädiktiven Regelung (auch Internal Model Control genannt) geschieht [49, 209].

Richalet et al. [91, 209] wenden die prädiktive Regelung zur Steuerung eines Roboters an. Sie unterscheiden die Soll-Trajektorie x_d und eine in jedem Abtastschritt k für N folgende Schritte neu festgelegte Referenztrajektorie x_r, die den Roboter von seinem aktuellen Zustand $x(k)$ auf die Soll-Trajektorie überführt. Die Referenztrajektorie wird so geplant, dass der Fehler bezüglich der Soll-Trajektorie wie ein vorgegebenes PT2-System abklingt. Weiterhin wird der Zustand des Roboters für den Fall, dass keine weiteren Stelleingriffe erfol-

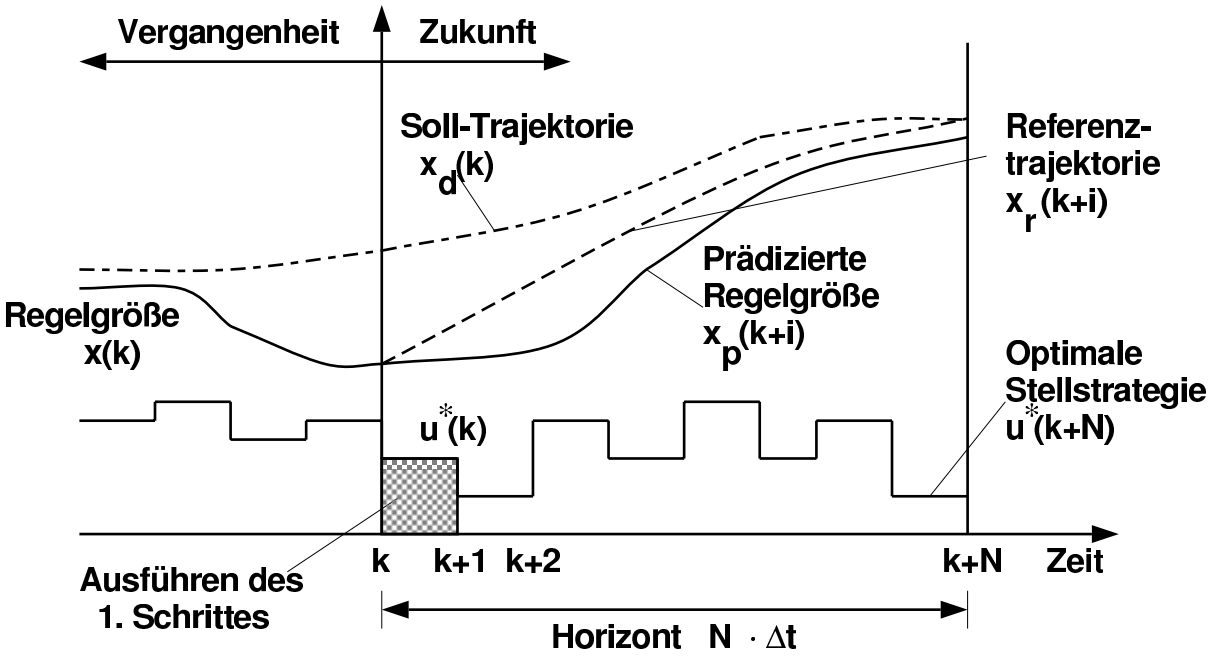


Abbildung 2.6: Prädiktive Regelung [110]

gen, voraus berechnet. Damit lassen sich durch Optimierung unter Verwendung eines einfachen Robotermodells die Kommandos u^* bestimmen, durch die der Roboter anstelle der voraus berechneten Trajektorie x_p die Referenztrajektorie abfährt. Das Kommando $u^*(k)$ für den ersten Abtastschritt ist schließlich die optimale Stellgröße, die nun abgegeben wird. Dies ist in Abb. 2.6 dargestellt.

Bei einer Variante von Jacubasch et al. [110] wird die Optimierung derart abgeändert, dass die Kommandos aus vorher definierten Basisfunktionen ausgewählt werden. Dadurch wird der Aufwand der Optimierung drastisch reduziert. Wörn und Postenrieder [260] verwenden eine lineare Reglergleichung, die neben den zukünftigen Werten der Referenztrajektorie auch die bisher prädizierten optimalen Kommandos und Messwerte sowie die Abweichung der vergangenen Messwerte von den vorhergesagten Werten gewichtet.

Hoyer et al. [106] verwenden keine Referenztrajektorie. Stattdessen wird die Differenz zwischen der vorhergesagten Position und der Soll-Position für den folgenden Abtastschritt durch den Regler in Abhängigkeit eines Gewichtungsfaktors minimiert.

In neuerer Zeit wird die prädiktive Regelung z. B. von Gangloff et al. [88, 89] verwendet. Dabei beziehen sie sich auf die unter dem Namen „Generalized Predictive Control (GPC)“ von Clarke et al. [49] vorgestellte Version, die aufgrund

der Impulsantwort der Regelstrecke einen linearen Regler entwirft (siehe auch [160, 197]).

2.2.3 Berücksichtigung von Elastizitäten

Bei Elastizitäten am Roboter muss unterschieden werden, ob die Nachgiebigkeit auf die Gelenke (Getriebe) beschränkt werden kann [6, 62, 113, 126, 143, 157, 237], oder ob die Arme selbst elastisch sind [45, 64, 161, 228, 263].

Beim Reglerentwurf wird die Ordnung des Systems durch den Ansatz von Nachgiebigkeiten erhöht. Meist ist die Messbarkeit des vollständigen Zustandsvektors vorausgesetzt, zumindest aber der Armwinkel [26, 143, 193]. Dagegen werden bei üblichen Robotersystemen aufgrund der besseren Auflösung nicht die Gelenkwinkel der Arme gemessen, sondern die Motorstellungen. Daher sind die meisten Verfahren bei Industrierobotern ohne Beobachter für die Armposition (siehe [113]) oder kartesische Messungen der Endeffektorposition [165, 228] nicht anwendbar. Ansätze, die die Armposition nicht benötigen [61, 157, 263], gibt es nur für einfache Roboter mit ein oder zwei Achsen. Lediglich Thümmel et al. [237] verwenden einen 6-achsigen Roboter. Aus Rechenzeitgründen beschränken sie sich auf den Computed Torque Ansatz, da dabei zur Berechnung der nichtlinearen Gleichungen nur die Soll-Bahn und deren zeitliche Ableitungen bekannt sein brauchen. Das erlaubt die Vorausberechnung einer Vorsteuerung.

Üblicherweise wird aber bei der Regelung von 6-achsigen Industrierobotern davon ausgegangen, dass die Roboter starr sind. Dies ist zulässig, da statische Positionsfehler sich beim Programmieren durch Teach-In nicht bemerkbar machen. Zur Vermeidung von dynamischen durch Elastizität bedingten Bahnabweichungen wird die Bandbreite der Roboterbewegung in der Praxis derart begrenzt, dass keine Schwingungen angeregt werden.

2.3 Modifikation von Bewegungen aufgrund von Sensorinformationen

Bei den beschriebenen Verfahren zur Regelung bahntreuer Roboterbewegungen wurde davon ausgegangen, dass die Bahn vorab definiert ist. In vielen Fällen gibt es aber bei der Bahnplanung eine gewisse Unsicherheit, die Modifikatio-

nen der ursprünglich geplanten Bewegung aufgrund von Sensorinformationen erfordert.

Bei Industrieroboteranwendungen erfolgt die sensorgestützte Bahnkorrektur, sofern sie überhaupt vorgesehen ist, durch Korrekturen der Lagesollwerte (vergl. Abb. 2.2). Dadurch kann man z. B. eine ganze Bahn kartesisch verschieben, weil das zu bearbeitende Werkstück nicht korrekt ausgerichtet ist.

Weiterhin kann man die Sensorschnittstelle im Takt der Robotersteuerung nutzen, um die jeweiligen Lagesollwerte (kartesisch oder in Roboterkoordinaten) zu definieren [137, 225]. Dabei muss aber mit Totzeiten und dynamischen Verzögerungen gerechnet werden.

2.3.1 Kraftregelung

Die ersten auf reinen Sensordaten basierenden Regelungsverfahren werteten Kraft- / Momentensensoren aus [266]. Die Differenzen zwischen der gemessenen Kraft und einer Soll-Kraft werden durch einen Kraftregeler in Geschwindigkeitssollwerte umgesetzt und erlauben mit einer standardmäßigen Kaskadenregelung die Einstellung einer gewünschten Kontaktkraft. Bei Addition einer festen Geschwindigkeit senkrecht zum Kraftvektor kann der Roboter einer Kontur folgen [77, 102, 152] oder eine Bahn auf einer Kontaktfläche beschreiben [256].

Eine Übersicht über neuere Ansätze zur Kraftregelung gibt [224]. In dieser Arbeit wird aber nur kurz auf die grundlegenden Konzepte eingegangen.

Mason [167] unterscheidet die bahntreue Regelung von Robotern in einem kraftgeregelten und einem positionsgeregelten Unterraum, wobei in Letzterem keine Kontaktkräfte aufgebracht werden können. Daraus ergibt sich eine hybride Regelung für Kraft und Position [30, 65, 127, 164, 207], die in Abb. 2.7 dargestellt ist.

Dabei sind die Kraftmesswerte $\mathbf{F}$, die Geschwindigkeit $\dot{\mathbf{x}}$ und die Position $\mathbf{x}$ in dem kartesischen System angenommen, in dem auch die Unterscheidung der Unterräume erfolgt. Dieses System nennt Khatib [127] „task frame“. Eine Selektionsmatrix $\mathbf{\Sigma}$ wählt die kraftgeregelten Komponenten aus, um die Sensorwerte mit den entsprechenden Soll-Kräften $\mathbf{F}_d$ zu vergleichen. Entsprechend wird durch $\mathbf{I} - \mathbf{\Sigma}$ der positionsgeregelte Anteil der Messwerte und Sollwerte ausgewählt. Die Summe der aus den beiden Reglern berechneten Stellgrößen

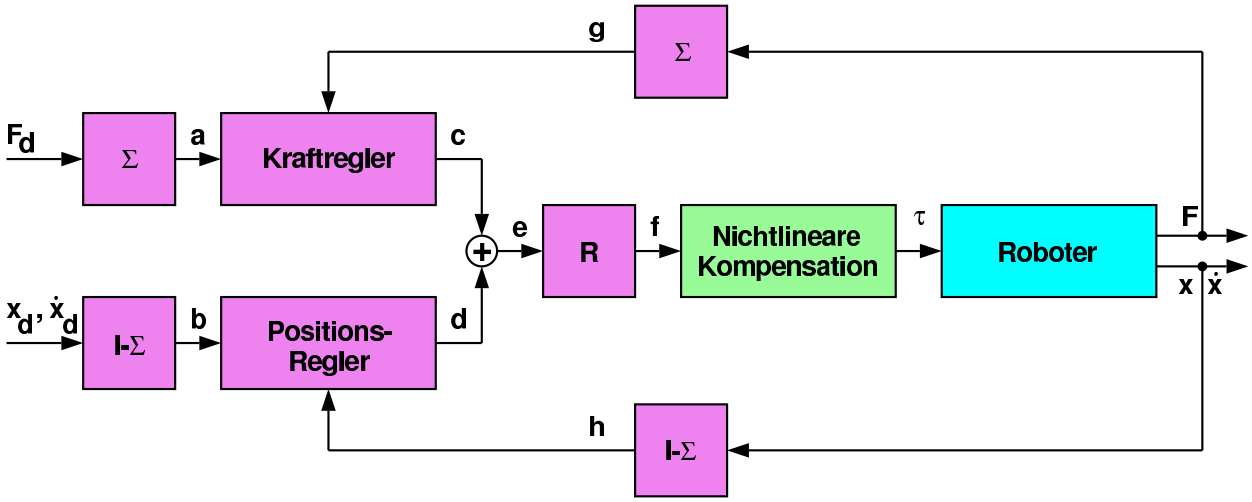


Abbildung 2.7: Hybride Regelung von Kraft und Position [221] a = ausgewählte Komponenten des Vektors der Soll-Kraft, b = ausgewählte Komponenten der kartesischen Lage- und Geschwindigkeitsvektoren, c = aufgrund des Kraftfehlers gewünschte Geschwindigkeit, d = aufgrund des Lage- oder Geschindigkeitsfehlers gewünschte Geschwindigkeit, e = gewünschte Geschwindigkeit im Task-Frame, f = gewünschte Geschwindigkeit in Basiskoordinaten, g = ausgewählte Komponenten der gemessenen Kraft, h = ausgewählte Komponenten der Lage und Geschwindigkeit, τ = Motormoment, $\mathbf{x}$ = kartesische Lage im Task-Frame, $\mathbf{x}_d$ = Soll-Lage, $\dot{\mathbf{x}}$ = kartesische Geschwindigkeit im Task-Frame, $\dot{\mathbf{x}}_d$ = Soll-Geschwindigkeit, $\mathbf{F}$ = gemessene Kraft im Task-Frame, $\mathbf{F}_d$ = Soll-Kraft, $\mathbf{R}$ Rotationsmatrix, $\mathbf{I}$ = Einheitsmatrix, $\mathbf{\Sigma}$ = Selektionsmatrix)

wird über die Rotationsmatrix $\mathbf{R}$ in Basiskoordinaten transformiert und an den Roboter abgegeben.

Hogan [104] verwendet dagegen im Kontaktfall den gleichen Regler wie für freie Bewegungen. Regelgrößen sind also nur die Position und die Geschwindigkeit. Im Kontaktfall erfolgt aber aufgrund der sensorisch erfassten Kräfte ein elastisches Ausweichen von der Soll-Trajektorie (programmierte Feder). Dabei können die Nachgiebigkeit und die Dämpfung der Ausweichbewegung vorgegeben werden. Für diesen Ansatz wird der Begriff Impedanzregelung verwendet (siehe Abb. 2.8) [82, 115, 185, 189, 223, 244].

Leider liefert ein Kraft-Momentensensor bei Montageaufgaben nicht immer ausreichende Informationen über die bei Fügevorgängen auftretenden Wechselwirkungen. Das kann einerseits daran liegen, dass der Endeffektor an verschiedenen Punkten Kontakt mit der Umgebung hat (Mehrpunktkontakt), was aufgrund des Messvektors allein nicht interpretiert werden kann. Andererseits misst ein Kraft- / Momentensensor nur im Kontaktfall, kann also benachbarte Objekte nicht wahrnehmen. Daher ist es manchmal sinnvoll, bei Montageaufgaben eine

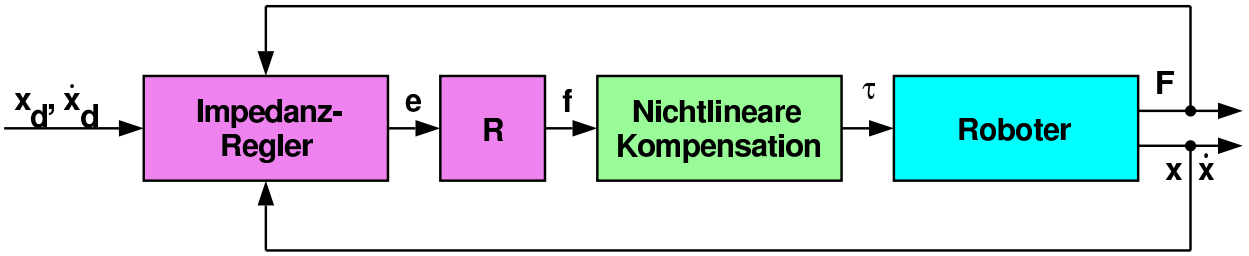


Abbildung 2.8: Impedanzregelung für Kraft und Position (Bezeichnungen wie bei Abb. 2.7)

heuristische Suchbewegung auszuführen, sofern eine geeignete Umweltmodellierung nicht vorhanden ist [158, 190]. Bei bekannter Umgebung ist dagegen meist eine modellgestützte Bestimmung des Montagezustandes möglich, wie in [174].

2.3.2 Andere sensorgestützte Regelungen

Abgesehen von durch unvollständige Beobachtung bedingten Mehrdeutigkeiten bezüglich der Umgebung lassen sich aus Sensordaten meist direkt Lagesollwerte oder Geschwindigkeitssollwerte ableiten. Typische Beispiele solcher Sensoren sind Entfernungsmesser und CCD-Kameras [35]. Im Gegensatz zu Kraft- / Momentensensoren sind sie positionsmäßig kalibriert, was die Regelung bezogen auf ein Referenzsystem erleichtert.

Normalerweise wird zum Zeitpunkt k eine Änderung der kommandierten Position $\mathbf{x}_c(k)$ in kartesischen Koordinaten bestimmt, die sich in Abhängigkeit der Differenz aus den aktuellen Sensorwerten $\mathbf{s}(k)$ und den Sensorsollwerten $\mathbf{s}_d(k)$ ergibt. Dabei sind neben dem allgemeinen Ansatz

$$\mathbf{x}_c(k) = \mathbf{x}_c(k-1) + \sum_i \mathbf{R}_i \cdot (\mathbf{s}_d(k-i) - \mathbf{s}(k-i)) \tag{2.2}$$

bei konstanten Sensorsollwerten insbesondere PD-Regler üblich.

$$\mathbf{x}_c(k) = \mathbf{x}_c(k-1) + \mathbf{R}_P \cdot (\mathbf{s}_d - \mathbf{s}(k)) - \mathbf{R}_D \cdot (\mathbf{s}(k) - \mathbf{s}(k-1)) \tag{2.3}$$

Die Matrizen $\mathbf{R}_{..}$ enthalten die Verstärkungsfaktoren und haben daher die Form einer Diagonalmatrix. Wenn die Sensordaten und die Bewegungsbefehle in unterschiedlichen Koordinatensystemen definiert sind, enthalten die Matrizen au-

ßerdem noch eine entsprechende Rotationskomponente. $\mathbf{R}_P$ und $\mathbf{R}_D$ gewichten den P-Anteil bzw. den D-Anteil des PD-Reglers.

Bei der Verarbeitung der Sensordaten gibt es ein grundlegendes Problem, das als Raum-Zeit-Problem (Hirzinger [102]) bezeichnet wird. Es besagt, dass eine sensoriell ermittelte Abweichung der Position von der Soll-Position nicht sofort ausgeglichen werden kann, ohne unendlich hohe Beschleunigungen aufzubringen. Dieses Problem tritt bei Kraftregelungen nur in abgeschwächter Form auf, da dort die Lagedifferenzen sehr klein sind, sodass sie meist innerhalb eines Abtastschrittes überwunden werden können. Bei größeren Abweichungen muss der Regler eine Bahn bestimmen, mit der das Ziel erreicht werden kann (siehe z. B. [31]).

Das Problem kann umgangen werden, indem die Regelung den Roboter nicht schnellstmöglich auf die Soll-Trajektorie zurückführt, sondern auf eine suboptimale Referenztrajektorie, die stetig von der aktuellen Lage ausgeht und die Soll-Trajektorie durch ein einfaches Gesetz erreicht [110, 260]. Diese Referenztrajektorie kann implizit auch durch ein Filter bzw. einen Regler für die Regeldifferenz implementiert sein [59, 203].

Diese Maßnahmen tragen dazu bei, dass die Stellgrößenbegrenzungen nicht erreicht werden, sie begrenzen aber gleichzeitig die erreichbare Bahntreue. Eine konkrete Verbesserung der Regelgüte lässt sich nur dann herbeiführen, wenn ein prädiktiver Sensor verwendet wird, der nicht die Regelgröße selbst misst, sondern eine Vorhersage der zu erwarteten Abweichungen der Werkzeugspitze von der sensoriell erfassten Bahn erlaubt. Dies kann ein Sensor sein, der in einiger Entfernung von der Werkzeugspitze in Bewegungsrichtung positioniert ist. Dabei wird angenommen, dass die Abweichungen zwischen kommandierten und erreichten Lagen des Roboters im Vergleich zu der Bahnunsicherheit gering sind. Pritschow et al. [202] untersuchen die Vorteile eines vorlaufenden Sensors und berechnen die damit realisierbaren Geschwindigkeiten beim kraftgeregelten Abfahren von Konturen mit a priori unbekanntem Verlauf.

Tsuji et al. [241] verallgemeinern den Ansatz der Impedanzregelung auf allgemeine Sensorregelungen. Dabei wird in Abhängigkeit von Sensordaten „elastisch“ von der gegebenen Referenzbahn abgewichen.

2.3.3 Regelung auf der Basis von 2D Sensorik

Eine besondere Stellung unter den Sensoren nehmen 2D Sensoren ein. Dazu gehören neben flächenmäßig messenden Entfernungsmessern (Scannern) auch intensitätsbildbasierte visuelle Sensoren (CCD-Kameras). Bei ihnen liegen im Gegensatz zu den bei Entfernungsmessern oder Scannern explizit vorliegenden Abstandsdaten lediglich Pixelwerte vor, die nicht direkt als kartesische Soll-Positionsänderungen interpretiert werden können. Stattdessen ist eine Vorverarbeitung der Intensitätsbilddaten erforderlich, nicht zuletzt aus Gründen der Datenreduktion. Ein visueller Sensor wird also durch ein gesamtes Bildverarbeitungssystem gebildet, das die Variablen $\mathbf{s}(k)$ aus Gleichung (2.2) oder (2.3) bestimmt.

Als typische Aufgabe für online durch 2D Sensoren unterstützte Regelungen wird in dieser Arbeit nur der Ansatz mit einer am Roboter angebrachten Kamera betrachtet. Dazu werden neben hybriden Methoden wie [53] sowohl bildbasierte als auch positionsbasierte Verfahren (Abb. 2.9 und Abb. 2.10) eingesetzt [37].

Bei positionsbasierten Verfahren [16, 36, 89, 166, 182] werden die in der Bildvorverarbeitung (Merkmalsextraktion) bestimmten Bildkoordinaten $\mathbf{p}$ der Merkmale genutzt, um eine 3D-Positionsdifferenz oder 6D-Lageänderung $\mathbf{\Delta x}$ zu schätzen, um die sich der Roboter bewegen soll. Dabei werden Modellannahmen benötigt, z. B. die Entfernung zwischen der Kamera und den Objektpunkten. Ein einfaches Regelgesetzes kann dann unter Verwendung der in der Robotersteuerung bekannten Rückwärtstransformation der Roboterkinematik Bewegungsbefehle $\mathbf{q}_c$ generieren. Regelung und Umrechnung der Merkmalspo-

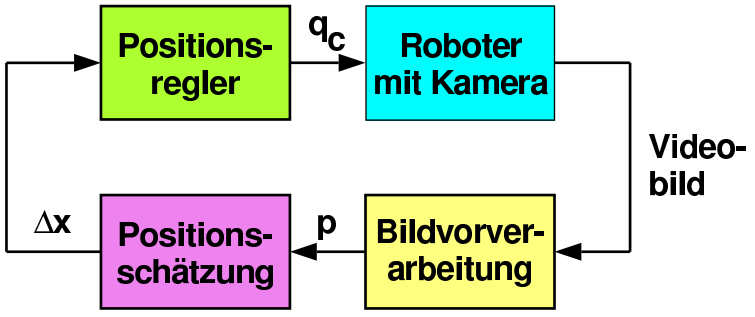


Abbildung 2.9: Positionsbasierte Regelung (dynamic position-based look-and-move) ($\mathbf{p}$ = Positionen der Merkmale im Raum der Bildkoordinaten, $\mathbf{\Delta x}$ = 3D-Positionsdifferenz bzw. 6D-Lageänderung, $\mathbf{q}_c$ = Bewegungsbefehl in Roboterkoordinaten)

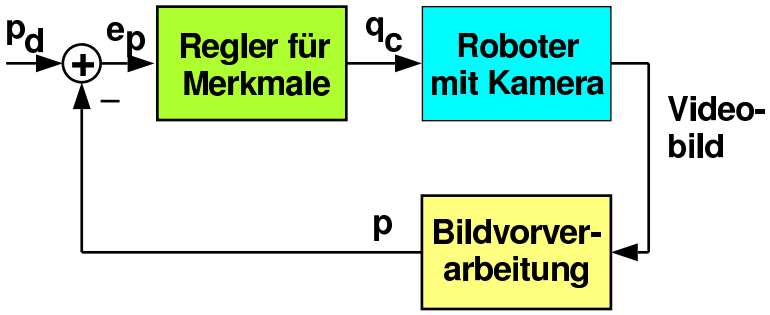


Abbildung 2.10: Bildbasierte Regelung (dynamic image-based look-and-move) [107] ($\mathbf{p}_d$ = Soll-Positionen der Merkmale im Raum der Bildkoordinaten, $\mathbf{e}_p$ = Merkmalsregeldifferenz, übrige Bezeichungen wie bei Abb. 2.9)

sitionen in Raumkoordinaten sind also getrennt. Bei Anwendung von Gleichung (2.2) für den Regler stellt $\mathbf{s}_d(k-i) - \mathbf{s}(k-i)$ die Positionsdifferenzen $\mathbf{\Delta x}$ im Raum dar.

Bei bildbasierten Verfahren [96, 99, 171, 196] werden dagegen die in Bildkoordinaten ermittelten Merkmalspositionen $\mathbf{p}$ direkt mit den Soll-Positionen $\mathbf{p}_d$ der Merkmale (z. B. der Bildmitte) verglichen. Der Funktionsblock „Regler für Merkmale“ in Abbildung 2.10 enthält neben der Regelung mit den Bildkoordinaten als Eingang auch die Aufgabe der Transformation auf das Koordinatensystem der Bewegungsbefehle (Hand-Auge-Transformation[1] [11, 248, 271]). Dazu wird i. A eine Jacobi-Matrix (Bildjacobi $\mathbf{J}_I$) gebildet, die Merkmalspositionen auf Positionsänderungen in kartesischen Weltkoordinaten oder in Roboterkoordinaten abbildet, also z. B.

$$\Delta \mathbf{x}_c = \mathbf{J}_I \cdot (\mathbf{p}_d - \mathbf{p}) \tag{2.4}$$

Bei Anwendung eines PD-Regelgesetzes in kartesischen Koordinaten ergibt sich daraus Gleichung (2.3). In dem Fall sind $\mathbf{s}$ und $\mathbf{p}$ identisch. Die Verstärkungsmatrizen $\mathbf{R}_P$ und $\mathbf{R}_D$ sind dabei nicht diagonal. Sie berücksichtigen auch die in der Bildjacobi enthaltene Information der Orientierung des Kamerasystems im Raum und des Abstands zum Objekt.

In weitergehenden Ansätzen werden anstelle der kartesischen Bewegungsbefehle $\mathbf{x}_c$ Kommandos in Roboterkoordinaten $\mathbf{q}_c$ oder direkt die Motormomente

[1]Der Begriff Hand-Auge-Transformation wird sowohl für die (konstante) Lage der Kamera bezüglich des Endeffektors (Hand) verwendet (bei Kommandierung des Roboters in kartesischen Endeffektorkoordinaten) als auch für die lageabhängige Transformation zwischen Bildkoordinaten einer ggf. feststehenden Kamera und Roboterkoordinaten (Gelenkwinkeln)

τ bestimmt. Bei diesen Ansätzen bildet die Bildjacobi-Matrix also auch die kinematische Rückwärtstransformation des Roboters ab. In diesem Fall ist die Bildjacobi stark positionsabhängig. Bei direkter Bestimmung der Motormomente ist weder ein Positionsregelkreis noch die Messbarkeit der Position durch die Robotersteuerung vorausgesetzt.

Bildbasierte Verfahren sind unempfindlicher gegenüber ungenauer Kenntnis der Roboterkinematik oder der Transformation zwischen Endeffektor und Kamera und eignen sich deshalb meist besser [107] zum Verfolgen bewegter Zielobjekte.

Verfahren zum Verfolgen eines Zielobjekts unter Berücksichtigung dynamischer Aspekte schätzen die Zielgeschwindigkeit und verwenden sie zur Vorsteuerung [46, 47, 52, 251]. Bei zusätzlicher Berücksichtigung der Roboterdynamik oder weiterer, positionsbasierter Sensordaten ist eine Transformation der Roboterbewegung in den Bildbereich [21, 36, 51] (bei bildbasierten Verfahren) oder die Umrechnung der Merkmale von Bildkoordinaten in 3D Roboterkoordinaten (bei positionsbasierten Verfahren) nötig. [252]

2.4 Mögliche Adaptionsverfahren

In den vorherigen Abschnitten dieses Kapitels wurden Roboterregelungen betrachtet. In diesem Abschnitt wird nun gezeigt, welche Möglichkeiten es zur Adaption der Regler an Prozessvariationen gibt. Dazu werden zunächst die wichtigsten Adaptionsverfahren beschrieben.

2.4.1 Strukturen von Adaptionsverfahren

Isermann unterscheidet zwischen adaptiven Reglern mit Referenzmodell (MRAS = model reference adaptive system) und adaptiven Reglern mit Identifikationsmodell (MIAS = model identification adaptive system) [108]. Im ersten Fall geht es darum, einem geschlossenen Regelkreis das Verhalten eines Referenzmodells aufzuprägen, z. B. die Antwort eines linearen Systems 2. Ordnung (Abb. 2.11). Im zweiten Fall wird die Regelungsstrecke zunächst identifiziert. Aufgrund der so gewonnenen Prozessparameter werden dann die Reglerparameter angepasst (Abb. 2.12). Dabei werden die üblichen Entwurfsverfahren für Regler zugrunde gelegt.

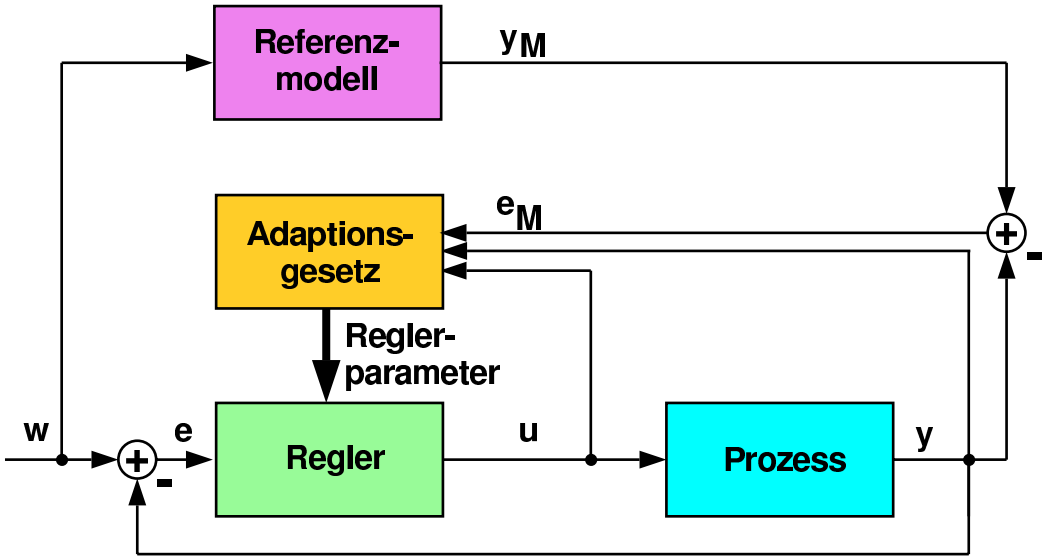


Abbildung 2.11: Adaption mit Referenzmodell [108]

Beide Ansätze werden sowohl mit parametervarianten als auch mit strukturvarianten Modellen, sowohl zeitdiskret als auch kontinuierlich realisiert. Man spricht daher auch von Parameteradaption und Strukturadaption.

Eine erschöpfende Beschreibung der möglichen Identifikationsverfahren oder der anwendbaren Reglerentwurfsmethoden würde den Umfang dieser Arbeit sprengen. Daher sei auf die entsprechende Literatur [84, 108, 109, 132, 243] verwiesen.

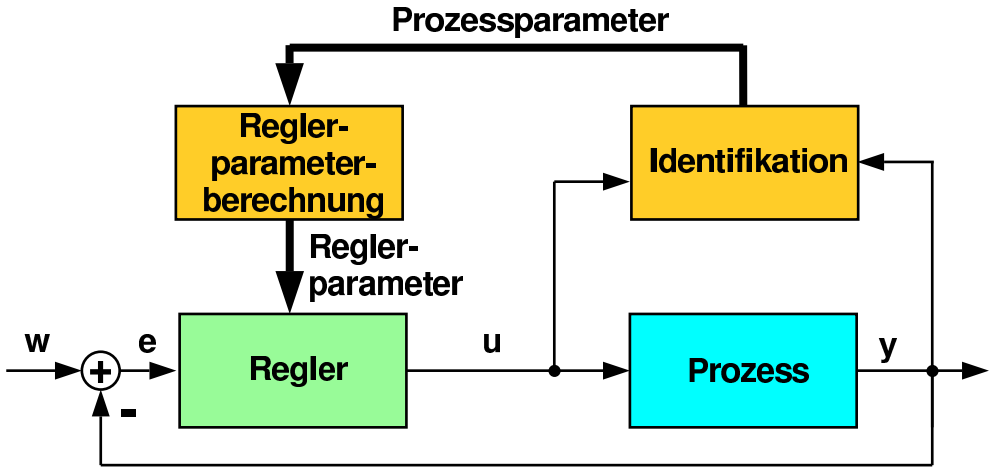


Abbildung 2.12: Adaption mit Identifikationsmodell [108]

2.4.2 Strukturadaptive Verfahren

Eine besondere Rolle spielen strukturadaptive Verfahren[2] Als Vertreter dieser Gruppe sind insbesondere Varianten von Künstlichen Neuronalen Netzen (KNN) und Neuro-Fuzzy Systemen zu nennen. Bei Neuronalen Netzen wird irrtümlicherweise nicht von adaptiven Systemen sondern von Lernsystemen gesprochen. Dieser Sicht soll nicht gefolgt werden, da bei den in dieser Arbeit behandelten Problemen kein explizites Lernen stattfindet, sondern konkret ein Adaptionsprozess vorliegt.

Genau genommen handelt es sich bei Unscharfen (Fuzzy) Reglern um abschnittsweise lineare (also parametrische) Systeme. Insgesamt ergibt sich ein auf Kennlinien basierender nichtlinearer Regler, dessen linearisiertes Verhalten stetig zwischen den verschiedenen Arbeitspunkten variiert. Die Unschärfe wird durch eine so genannte Zugehörigkeitsfunktion repräsentiert, die einer Parametrisierung der Nichtlinearitäten entspricht. Ein Neuro-Fuzzy Regler ist ein adaptives Fuzzy System [265]. Ein Überblick zu Ansätzen der Unscharfen Regelung ist in der Literatur [92, 128] zu finden.

Bei der Wahl der Struktur und der Lernparameter Neuronaler Netze als Regler geht es primär darum, vorgegebene Funktionen zur Abbildung von Eingangsvektoren auf Ausgangsvektoren korrekt nachzubilden. Hierzu werden die Netze mit Musterfunktionen als Beispielen trainiert. Dabei wird üblicherweise der Backpropagation Algorithmus [212] verwendet. Andere Lernverfahren sind in Abschnitt 2.4.3 erläutert. Dem im Vergleich zu parameteradaptiven Repräsentationen aufwendigen Training steht ein großes Potential an Möglichkeiten entgegen, auch stark nichtlineare Funktionen darzustellen. Dies rechtfertigt durchaus den Einsatz von Neuronalen Netzen.

In dieser Arbeit werden Neuronale Netze zur Repräsentation von Funktionen mit mehreren Eingängen im Rahmen von Adaptionsverfahren verwendet und nicht als selbständige Verfahren. Im Folgenden sollen die wichtigsten in der Literatur beschriebenen KNN-Verfahren zum Entwurf und zur Realisierung adaptiver Regelungsverfahren kurz vorgestellt werden, da die Adaptionsalgorithmen der unterlagerten Verfahren auch unabhängig von der konkreten Repräsentation Bedeutung haben.

[2]Strukturadaptive Verfahren werden auch ausführungsadaptive (performance adaptive) Systeme genannt [238].

Die gewünschte Abbildung wird approximiert, indem mehrere einfache Funktionen überlagert werden. Möglich sind entweder lokale Funktionen, wie bei Netzwerken mit radialen Basisfunktionen [198], oder globale Nichtlinearitäten wie bei mehrschichtigen Perzeptrons [212].

Im einen Fall überlagert man Funktionen, die nur in eng begrenzten Bereichen des Eingangsraumes von null verschiedene Funktionswerte liefern. Diese Funktionen beschreiben jeweils Gauß'sche Glockenkurven

$$y = a \cdot e^{-b \cdot (\mathbf{x} - \mathbf{x}_0)^2}. \tag{2.5}$$

Im anderen Fall beeinflusst jede Teilfunktion die Funktionswerte in einem weiten Bereich des Eingangsraumes. Dabei werden Aktivierungsfunktionen wie

$$y = tanh(x') = \frac{e^{x'} - e^{-x'}}{e^{x'} + e^{-x'}} \tag{2.6}$$

angewendet. Die Eingänge x' der Aktivierungsfunktionen werden als gewichtete Summen der Elemente des Eingangsvektors $\mathbf{x}$ der Abbildung berechnet.

Wegen der Symmetrie von Gleichung (2.6) wird oft auch die Sigmoid Funktion verwendet:

$$y = \frac{1}{1 + e^{-x'}} \tag{2.7}$$

Beide Netztypen haben Vor- und Nachteile. Bei globalen Nichtlinearitäten ist das Training weitläufiger Abbildungen aufwendiger, da jede Trainingsinformation sich auf das gesamte mehrschichtige Netz auswirkt. Andererseits erfordern lokale Nichtlinearitäten in der Nähe eines Trainingspunktes neben der Adaption der Verstärkungen auch die Anpassung der Kurvenform und -lage, also der Parameter a, b, und $\mathbf{x}_0$ in Gleichung (2.5). Dagegen ergeben sich die Werte x' der einzelnen Neuronen durch lineare Gleichungssysteme, deren Parameter relativ einfach adaptiert werden können.

Durch Verallgemeinerung ergeben sich bei globalen Nichtlinearitäten in untrainierten Regionen unter Umständen grob falsche Funktionswerte. Dagegen liefern lokale Nichtlinearitäten in untrainierten Regionen keine von null verschie-

denen Funktionswerte.

Schließlich unterschieden sich die KNNs noch durch ihre Repräsentationsfähigkeit. So haben lokale glockenförmige radiale Basisfunktionen z. B. Probleme mit der Darstellung abschnittsweise konstanter Funktionen. Erweiterungen des Ansatzes von Gleichung (2.5) sind ferner nötig, wenn die Rotationssymmetrie zur Abbildung hinderlich ist. Andernfalls ist eine sehr große Zahl an Neuronen nötig, um eine hinreichend genaue Abbildung zu erzeugen. Dies gilt wegen des exponentiellen Aufwands insbesondere bei hochdimensionalen Abbildungen.

Andere lokale Repräsentationen sind CMAC [7] nach Albus, ein assoziatives Abbildungsmodell in Anlehnung an Abbildungsstrukturen im Kleinhirn, oder NPPL [122] bzw. LOLIMOT [186], zwei abschnittsweise definierte, an den Übergängen differenzierbare Funktionsansätze. Diese Verfahren versuchen, den Speicherplatzbedarf im Vergleich zum abgebildeten Raum klein zu halten. Insbesondere beim CMAC bedeutet dies aber Kompromisse, die zur Vermeidung von Darstellungsgranularität lokal variable Auflösungstopologien erfordern.

Bei den globalen Darstellungen geht man von mehrschichtigen Neuronen aus. Die Ausgabe eines Funktionswertes erfolgt dabei wie folgt (siehe auch Anhang B): Zunächst wird in jedem Neuron der ersten verdeckten Schicht die gewichtete Summe aller Eingänge gebildet. Dann wird der Ausgang dieser Neuronen bestimmt, indem die Aktivierungsfunktion auf diese gewichtete Summe angewendet wird. Diese Vorgehensweise wird nun schichtenweise bis zu den Ausgangsneuronen ausgeführt. Dabei enthalten die Gewichtungen die Information, die vorher gelernt wurde. Zum Erlernen der günstigsten Gewichtungen können unterschiedliche Ansätze wie Backpropagation, Reinforcement Learning oder Q-Lernen [20, 212, 253] angewandt werden.

Meist werden die Neuronen mit Schwellwertparametern ausgestattet, d. h. sie enthalten in jeder Schicht einen zusätzlichen lernbaren Parameter. Dieser kann als zusätzliches Gewicht betrachtet werden, das aber nicht einen Wert aus der nächst niedrigeren Schicht gewichtet, sondern die Konstante eins (siehe Gleichung (B.4)).

Die Aktivierungsfunktion muss nichtlinear sein, da sich sonst nur lineare Funktionen darstellen lassen. Grundsätzlich geeignet sind auch Funktionen, bei denen sich der Ausgang nur in einem beschränkten Bereich der Eingangsvariablen merklich verändert. In der klassischen Darstellung des Perzeptrons [211] wird deshalb die Stufenfunktion

$$y = \begin{cases} 0 & x' \leq 0 \\ 1 & x' > 0 \end{cases} \tag{2.8}$$

verwendet. Zur Darstellung kontinuierlicher Funktionen sind dagegen mehrfach differenzierbare unsymmetrische Funktionen wie die Gleichungen (2.6) und (2.7) vorteilhafter. In manchen Netzen werden nur die verdeckten Schichten mit nichtlinearer Aktivierungsfunktion angesetzt.

Mehrschichtige Netze werden klassisch durch den von Rumelhart et al. [212] beschriebenen Backpropagation Algorithmus trainiert. Die Herleitung hierzu verwendet den Ansatz der Berechnung der Ausgangswerte (forwardpropagation) und bildet die partiellen Ableitungen von Änderungen der Ausgangswerte bezüglich der Gewichte. Dies geschieht rückwärts vom Ausgangsneuron aus durch Anwendung der Kettenregel und der Produktregel, die die Einflüsse der Eingänge der einzelnen Neuronen von den Gewichten trennt. Es ergibt sich eine um den aktuellen Zustand linearisierte Korrektur, die mit einem Lernfaktor gewichtet wird. Dabei werden die Ausgangsfehler auf die Parameter zurückgekoppelt. Dies ist ein Adaptionsansatz, der auch zur Regleradaption anwendbar ist (siehe Abb. 2.13).

Dabei bildet der Regler das inverse dynamische Modell, während das adaptierende Modell das direkte dynamische Modell darstellt, aufgrund dessen der Regler adaptiert wird. Je nach Implementierung des Modells erfolgt die Inversion des Modells entweder durch ein Suchverfahren, also durch Ausprobieren von verschiedenen Stellgrößen und Vergleich der vom Streckenmodell gelieferten Aus-

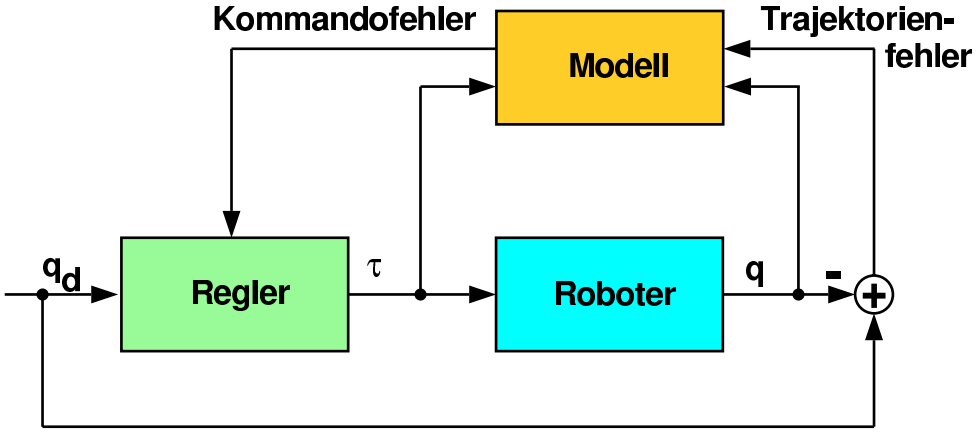


Abbildung 2.13: Adaption des Reglers aufgrund eines Streckenmodells (auch forward and inverse modeling) [124] (Regler und Modell können durch KNNs realisiert sein.)

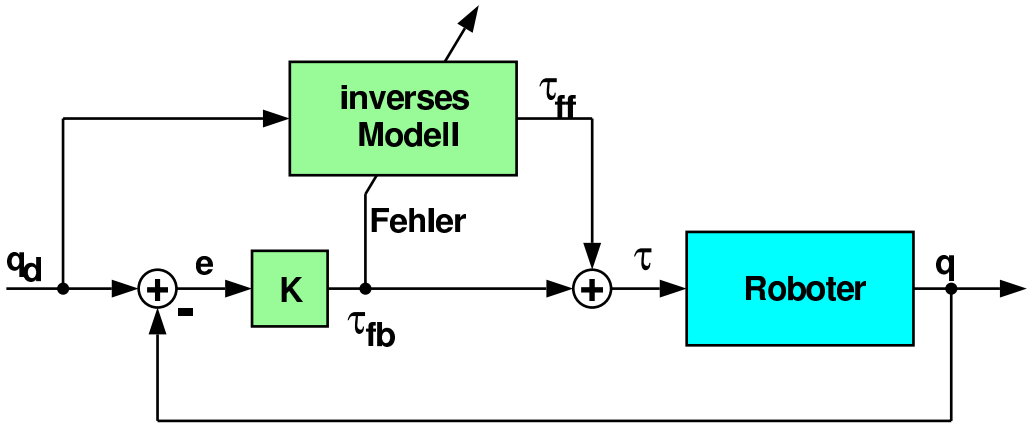


Abbildung 2.14: Feedback Error Learning (auch adaptive computed torque oder adaptive zentrale Vorsteuerung) [124]

gangswerte [77], oder durch Rückwärtspropagierung der Regeldifferenzen durch das Modell [114]. Der Unterschied zur allgemeinen adaptiven Struktur MIAS (Abb. 2.12) besteht darin, dass dort nur Modell*parameter* zum Regler*entwurf* ausgelesen werden.

Eine andere Art des Zurück-Propagierens ist das so genannte feedback error learning (Abb. 2.14). Dabei wird nicht die Regelstrecke, also das Robotermodell adaptiert, sondern das Lernen des inversen Modells beginnt beim Streckeneingang. Dazu kann ein einfacher Regler verwendet werden.

Ein andere Interpretation dieses Lernvorgangs führt zu dem Schluss, dass das unterlagerte Adaptionsgesetz mit dem Regelgesetz identisch ist.

Im allgemeinen Fall wird man aber einen anderen Adaptionsalgorithmus verwenden. Dabei gibt es sogar die Möglichkeit einer nichtkausalen Adaption [117], also z. B.

$$\begin{aligned} \tau_{ff,neu} &= \tau_{ff} + L_{-1} \cdot z^{-1} \cdot e + L_0 \cdot e \\ &+ L_1 \cdot z \cdot e + L_2 \cdot z^2 \cdot e + L_3 \cdot z^3 \cdot e. \end{aligned} \tag{2.9}$$

Hier beschreibt z^i den Operator der zeitlichen Verschiebung um i Abtastschritte. Wenn beim Abfahren einer Bahn das Moment $\tau_{ff}(k)$ in den auf den Schritt k folgenden Abtastschritten $k+i$ die Regeldifferenzen $e(k+i)$ erzeugt hat, kann durch geeignete Wahl der Parameter L_i erreicht werden, dass das inverse Modell beim nächsten Abfahren der Bahn ein optimales Moment $\tau_{ff,neu}(k)$ abgibt, das

bei $\tau_{fb}(k) = 0$ auch die folgenden Regeldifferenzen verschwinden lässt.

Verschiedene Autoren [63, 180] weisen auf mögliche Konvergenzprobleme der Adaption nach Gleichung (2.9) hin, da dort die Stellgröße integrierend zurückgeführt wird. Deshalb empfehlen sie die allgemeine Adaptionsregel

$$u_{neu}(z) = F(z) \cdot u(z) + L(z) \cdot e(z), \tag{2.10}$$

wobei sowohl $L(z)$ als auch $F(z)$ Funktionen entsprechend Gleichung (2.9) repräsentieren. u steht je nach Anwendung für eine vorgesteuerte Position oder ein vorgesteuertes Moment. e ist die Regeldifferenz, die sich aus der Differenz zwischen den Sollwerten w (z. B. den Gelenksollwerten $\mathbf{q}_d$) und den Messwerten y (z. B. den Gelenkvariablen $\mathbf{q}$) ergibt. Bei einer Streckenübertragungsfunktion $G(z)$ des geschlossenen Regelkreises nach Abb. 2.14 konvergiert die Adaption $e(z) = w(z) - G(z) \cdot u(z) = \mathbf{q}_d - G(z) \cdot \tau_{ff}$, wenn

$$|F - L \cdot G| < 1 \tag{2.11}$$

für alle z erfüllt ist. Bei $F = 1$ führt eine Phasendrehung mit $L \cdot G < 0$ also unabhängig von der Amplitude zum Verlust der Konvergenz.

Der ideale Lernoperator ergibt sich aus der Regelstrecke selbst. D. h. man bildet ein Modell der Strecke und verwendet dieses Modell invers als Adaptionsregel für den Regler selbst [262]. Dieser Ansatz ist auch als indirektes Lernen bzw. forward and inverse modeling bekannt. Diese Struktur ist in Abb. 2.13 dargestellt.

Eine nach den Strukturen aus Abb. 2.14 und Abb. 2.13 dritte Möglichkeit zur Adaption des inversen Modells besteht in der direkten Verwendung des Streckenmodells selbst [240]. Dazu ist es hilfreich, wenn man das Modell direkt invers bestimmen kann, z. B. durch Training eines Assoziativspeichers [7, 15, 176] oder eines Neuronalen Netzes [217], bei dem der Streckenausgang als Eingang und der Streckeneingang als zu trainierender Soll-Ausgang dient. Dies wird auch als time inversion training oder direct inverse modeling bezeichnet. Die Adaption des inversen Modells ist in Abb. 2.15 dargestellt. Bei der Regelung wird das inverse Modell wie der Regler in Abb. 2.13 verwendet.

Die Strukturen aus Abb. 2.13 und Abb. 2.15 unterscheiden sich also dadurch,

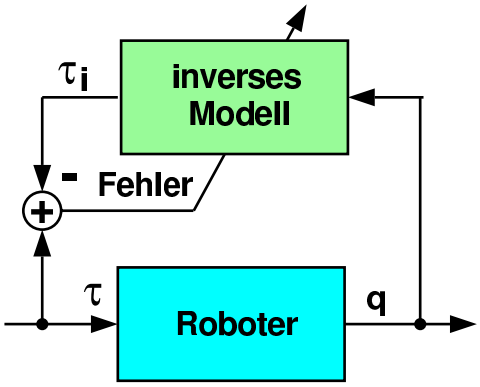


Abbildung 2.15: Adaption durch Zeitinversionstraining (auch direct inverse modeling) [124]

dass es im einen Fall zwei zu adaptierende Funktionsblöcke gibt, während im anderen Fall beide Funktionsblöcke identisch sind.

Bei getrennt realisiertem Regler und Modell kann der Regler im Gegensatz zur Struktur mit nur einem zu adaptierenden Funktionsblock inkrementell verbessert werden, sodass im Endeffekt Regler und Modell nicht exakt invers zueinander sind. Dies kann Vorteile bringen, wenn entweder die Inversion des Modells nur linearisiert erfolgen kann, wie beim Rückwärtspropagieren von der Regeldifferenz zum Kommando [114], oder wenn das Modell strukturbedingt ungenau ist. So erlaubt selbst ein Modellfehler von 10 % bei inkrementellem Training einen geringen Fehler der Regelung, da die verbleibenden Regelfehler wieder zurückgeführt werden.

Ein weiteres Problem des Zeitinversionstrainings ist es, dass ein inverses Modell im Gegensatz zu einem vorwärtsgerichteten Modell nicht zielgerichtet trainiert werden kann [124]. Wenn ein Kommando τ eine Beschleunigung $\ddot{q}$ hervorruft, weiß man noch nicht, welches Kommando für die gewünschte Beschleunigung $\ddot{q}_d$ nötig ist. Dagegen liefern sowohl das Rückwärtspropagieren als auch das Suchverfahren ein neues Kommando, das dann am realen Prozess erprobt werden kann. Dadurch liefert ein durch Zeitinversionstraining adaptiertes inverses Modell im Gegensatz zu einem getrennt realisierten Regler erst nach umfassendem Training zufrieden stellende Kommandos [172].

Ein weiterer Nachteil des Zeitinversionstrainings ist, dass das Regelverhalten nicht wählbar ist, sondern direkt dem Modell entspricht. Dadurch ist es nicht möglich, Filterungen vorzusehen oder die Regelung auf andere Weise robust zu machen. Eingeschränkt gilt diese Aussage auch für das feedback error learning, da der Regler dort primär auf Stabilität der Rückkopplung ausgelegt sein muss,

wodurch ggf. keine Freiheitsgrade zur Adaption bleiben.

2.4.3 Lernverfahren

Die bisher beschriebenen Adaptionsverfahren verwenden ein identifiziertes Modell oder eine heuristisch vorgegebene Adaptionsregel zur Anpassung des Reglers. Dies ist bei einfachen Regelungsaufgaben möglich, nicht aber bei komplexen Systemen. In dem Fall sind Lernverfahren nötig.

Dabei unterscheidet man *überwachtes Lernen*, bei dem die Sollwerte durch einen Lehrer vorgegebenen werden, und *unüberwachtes Lernen*, bei dem das System selbsttätig klassifizieren muss. Das überwachte Lernen entspricht der Anpassung des Reglers anhand der Modellinformation. Unüberwachtes Lernen wird dagegen zu Regelungsaufgaben in der Regel nicht eingesetzt. Eine mögliche Anwendung ist die Einteilung von Eingabemustern in Klassen, in denen unterschieldiche Stellgrößen nötig sind. Die für die jeweilige Klasse erforderlichen Kommandos werden individuell vom Bediener gewählt.

Eine dritte Art des Lernens ist das Lernen durch Verstärkung (*reinforcement learning*, auch Lernen durch Rückmeldung). Dabei wird die Trainingsrückmeldung auf Hinweise oder Beurteilungen reduziert, z. B. auf ein binäres Signal (Erfolg / Misserfolg) nach Ausführung einer Trajektorie (siehe z. B. [19, 116, 233, 254]). Klassisches Beispiel hierzu ist das Balancieren eines inversen Pendels auf einem Wagen [20].

Dabei wird dem Lernsystem zwar nicht die ideale Stellgröße selbst mitgeteilt, wohl aber, ob die Reaktion (auf mehrere abgegebene Stellgrößen) gut oder schlecht ist. Das System lernt dann eine Korrektur der Stellgrößen, durch die eine möglichst große Verstärkung erfolgt.

Es gibt verschiedene Varianten des Lernens durch Verstärkung, z. B. eine zweistufige mit einem Modell, dem so genannten adaptive critic element, das das Verhalten in jedem Abtastschritt beurteilt. Dabei wird das adaptive Kritiker-Element wiederum nur aufgrund des Verstärkungssignals trainiert [268].

Vor etwa 10 Jahren wurden dann Algorithmen mit gesicherter Konvergenz entwickelt. Das sind das so genannte Q-Learning [253] und das Temporal-Difference Learning [232]. Dabei wählt man die Aktionen derart, dass Vorhersagen des Verstärkungssignals optimiert werden. Temporal-Difference Learning

lernt schneller, da beim Training die Änderung der Vorhersage betrachtet wird und nicht das wirkliche Verstärkungssignal, das erst später zur Verfügung steht. Dies ist ähnlich wie im Backpropagation Algorithmus, in dem die gewünschten Änderungen rückwärts weitergeleitet werden [19].

Nachteil des Lernens durch Verstärkung ist, dass die gute Lernfähigkeit durch einen hohen Trainingsaufwand erkauft werden muss. Lernverfahren werden daher meist nur dann für Regelungsaufgaben angewandt, wenn einfache Adaptionsverfahren scheitern.

2.4.4 Adaptionsverfahren bei Roboterregelungen

Einige der erwähnten Adaptionsverfahren werden heute bereits bei Robotern experimentell eingesetzt. Dabei werden entweder einfache PD-Reglers durch ein nichtlineares Element ersetzt [141] oder es werden modellbasierte Verfahren adaptiert. Beide in Abschnitt 2.2.1 beschriebenen Strukturen (Abb. 2.4 und Abb. 2.5) eignen sich zur Adaption des inversen Modells. Sie ergeben entweder eine adaptive modellbasierte Regelung [111, 123] nach Abb. 2.16 oder die bereits erwähnte adaptive zentrale Vorsteuerung (auch adaptive computed torque method oder feedback error learning [50, 105, 177]) nach Abb. 2.14. Daneben gibt es Mischformen, z. B. nach Kim und Lewis [129], bei denen die adaptiven Regler sowohl die Soll-Trajektorie als auch die Regeldifferenzen gewichten, oder nach Tomiyama et al. [239], bei denen adaptive Regler für die Rückkopplung mit Neuronalen Netzen für die Vorsteuerung kombiniert sind. Chen et al. [40] kombinieren einen vorgegebenen Computed-Torque Anteil mit einem adaptierten Fuzzy Regler, der $\mathbf{e}$ und $\dot{\mathbf{e}}$ gewichtet. Popescu et al. [201] fassen beide in Abb. 2.16 zu adaptierenden Funktionsblöcke in einem Neuronalen Netz mit den Eingängen $\ddot{\mathbf{q}}_d, \dot{\mathbf{e}}, \mathbf{e}, \dot{\mathbf{q}}, \mathbf{q}$ zusammen.

2.4.5 Steuerung für wiederkehrende Bewegungen

Da Industrieroboter für getaktete zyklische Bewegungsequenzen eingesetzt werden, beschäftigten sich die ersten Adaptionsverfahren mit dem Trajektorienlernen, also dem Lernen der Anweisungen für eine feste Trajektorie. Beim Trajektorienlernen wird die vorgegebene Bahn mehrfach abgefahren, wobei sich die Regelfehler sukzessive reduzieren. Die Strukturen entsprechen dabei denen aus

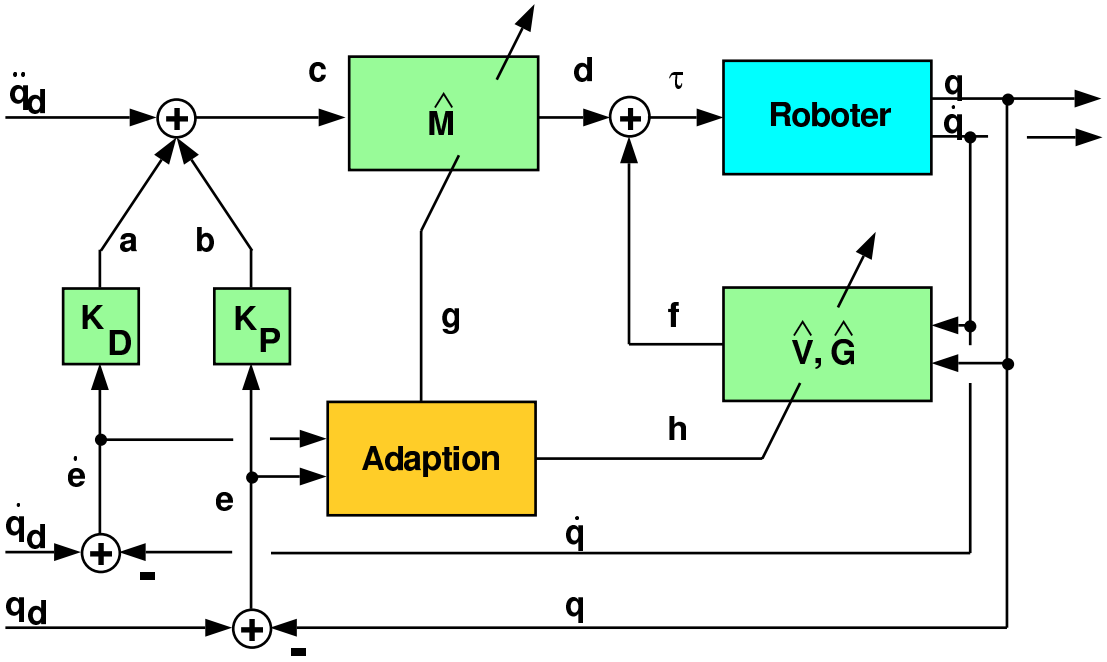


Abbildung 2.16: Adaptive modellbasierte Roboterregelung nach [57] ($\mathbf{K}_P$ Verstärkungsfaktor für P-Anteil des PD-Reglers, $\mathbf{K}_D$ Verstärkungsfaktor für D-Anteil des PD-Reglers, $\hat{M}$ = Modell der Massenmatrix M, $\hat{V}$ = Modell für den Einfluss der Coriolis- und Zentrifugalkräfte nach Gleichung (2.1), $\hat{G}$ = Modell für die auf den Arm wirkenden Gewichtskräfte G, a = Ausgang des D-Anteils des PD-Reglers, b = Ausgang des P-Anteils des PD-Reglers, c = aufgrund der aktuellen Regeldifferenz gewünschte Beschleunigung, d = Anteil des Motormoments zur Beschleunigung, e = Positionsregeldifferenz, f = Anteil des Motormoments zur Kompensation des Einflusses von Gravitation, Coriolis- und Zentrifugalkräften, g = Korrekturvorschrift für $\hat{M}$, h = Korrekturvorschrift für $\hat{V}$ und $\hat{G}$)

Abschnitt 2.4.4 [13, 55]. Für das Trajektorienlernen wird auch der Begriff „Iterativ Learning Control (ILC)“ verwendet [42, 43, 94, 181, 195], was hier aber nicht sinnvoll ist, da auch lernende Regelungen nach Abschnitt 2.4.3 iterativ arbeiten können. Dabei handelt es sich allerdings nicht um zyklische Bewegungen.

Bei Cheah und Wang [38] wird die Soll-Trajektorie durch eine Referenztrajektorie und eine Zielimpedanz gegeben (vergl. Abschnitt 2.3.1). Lucibello [162] definiert das gewünschte Verhalten durch Kontaktkräfte. Eine online Anpassung der Bahn bei unerwarteten Kräften erfolgt in beiden Fällen nicht, sodass es sich um ein reines Trajektorienlernen handelt.

Das Lernen einer festen Trajektorie ist einfacher als die Adaption allgemeiner Regelungen, da der Regler bzw. die Vorsteuerung nicht für den ganzen Arbeitsraum, sondern nur für eine feste Trajektorie trainiert werden müssen. Bei

zentraler Vorsteuerung reicht sogar die Speicherung der zeitlichen Folge der Ausgangswerte des inversen Modells. Dadurch ist die erreichbare Genauigkeit höher als beim Training mit allgemeinen Bewegungen, da die Speicher für das inverse Modell oder für die Stellfolge nicht die gesamte inverse Dynamik des Roboters repräsentieren können müssen, sondern nur die um die entsprechende Trajektorie linearisierte Funktion. Dies wurde auch von Burdet et al. [32] bestätigt.

Auch ein bereits oben diskutiertes Problem kann durch eine adaptive Steuerung beherrscht werden. So beeinflusst eine Stellgröße $u(k)$ zum Zeitpunkt k die zukünftigen Positionen $y(k+i)$ und muss daher in Abhängigkeit der zukünftigen Soll-Positionen $w(k+i)$ oder zumindest der Ableitungen $w^{(i)}(k)$ der aktuellen Soll-Position gewählt werden. Beim Trajektorienlernen ist dies immer möglich, da die gesamte Trajektorie bekannt ist. Bei der Adaption einer Vorsteuerung durch ein inverses Modell ist zwar eine nichtkausale Adaptionsregel möglich, die Reproduzierbarkeit des Trainierten ist aber nur dann gegeben, wenn als Eingang der Vorsteuerung auch die zukünftigen Sollwerte bzw. die Ableitungen des aktuellen Sollwertes zur Verfügung stehen.

Dagegen arbeiten reine Steuerungen nicht zufriedenstellend, wenn die Bewegungen nicht reproduzierbar sind, sei es aufgrund einer instabilen Strecke (z. B. Roboter ohne interne Positionsregelung) oder aufgrund von Störungen. In dem Fall muss die Steuerung für das geregelte System entworfen werden [179, 191] und die adaptive Steuerungskomponente muss die verbleibende Unsicherheit berücksichtigen. Alternativ wird zur (online) Adaption der Bewegungsanweisungen auch die jeweils aktuelle Regeldifferenz verwendet (siehe z. B. Chen et al. [44]), was einer Messwertrückführung entspricht.

2.5 Schlussfolgerungen

In diesem Kapitel wurden die wichtigsten Ansätze zur Regelung von Robotern vorgestellt.

Bei der Entscheidung zwischen modellgestützt berechneten Reglern auf der einen Seite und adaptiven bzw. lernenden Regelungsysstemen auf der anderen Seite sind die modellbasierten Verfahren kritisch zu bewerten, da meist kein hinreichend genaues Robotermodell zur Verfügung steht und die Identifikation

und Verifikation sehr aufwendig ist. Es kann allerdings angenommen werden, dass exakte Modellkenntnisse von Vorteil für den Reglerentwurf sind. In den folgenden Abschnitten wird aber nicht weiter darauf eingegangen.

Bei den adaptiven Systemen wird zwischen Reglern für allgemeine Bewegungen und dem Lernen einer speziellen Trajektorie unterschieden. Das Trajektorienlernen erscheint dabei als einfacher Sonderfall der allgemeinen Adaptionsaufgabe. Es wird daher nur als Zwischenschritt bei der Reglerbestimmung aufgefasst, zumal die praktische Bedeutung bei üblichen Roboteraufgaben gering ist.

Es werden also insbesondere adaptive Systeme zur allgemeinen Regelung betrachtet. Diese Auswahl ist geeignet für ständig wechselnde Aufgaben und unterschiedliche Roboter.

Bezüglich der Struktur der Adaption sind Ansätze vorteilhaft, die es erlauben, das gewünschte Verhalten des Systems zu spezifizieren. Dies ist bei den Methoden, die speziell für strukturadaptive Modelle entworfen wurden, meist nicht der Fall. Die Bedeutung Neuronaler Netze kann also nicht im direkten Lernen eines Reglers liegen.

Verbesserungen der Genauigkeit bei schnellen Bewegungen lassen sich insbesondere durch Betrachtung der zeitlichen Ableitungen der Soll-Positionen bzw. durch zukünftige Positionen erreichen. Dies entspricht prädiktiven Regelungsverfahren oder zentralen Vorsteuerungen (computed torque). Dieser Ansatz schließt ein, dass die Aufgabe dem Regler umfassender bekannt sein muss als dies bei bestehenden Robotersteuerungen der Fall ist. Bei sensorgestützt geplanten Bahnen erfordert dies eine Prädiktion der Sensorwerte.

Die Untersuchung der Genauigkeit von Industrierobotern beschränkt sich meist auf statische Betrachtungen. Die dynamische Genauigkeit, also die Abweichung von einer definierten Bahn, wird in der Literatur nicht betrachtet. Daher folgt in Kapitel 3 eine Untersuchung der Bahngenauigkeit von industriellen Systemen. Dabei wird auch der Einfluss von Elastizitäten bewertet, die bei den bisher vorgestellten Verfahren meist nicht berücksichtigt sind.

Für den Entwurf eines Steuerungsansatzes in Kapitel 4 folgt daraus, dass ein parameteradaptives Verfahren entwickelt wird, das als zusätzliche Hierarchiestufe zu einer der in Abschnitt 2.1 erläuterten und in der Praxis weit verbreiteten Roboterregelungen eingesetzt wird. Dabei können Aspekte der prädiktiven Regelung nach Abschnitt 2.2.2 übernommen werden. Sensorsignale werden entge-

gen den meisten in Abschnitt 2.3 beschriebenen Ansätzen nicht direkt sondern durch eine weitere Hierarchiestufe berücksichtigt.

Kapitel 3

Bewertung von industriellen Robotersystemen bezüglich ihrer Bahngenauigkeit

Sowohl die Literatur aus Kapitel 2 als auch die Datenblätter der meisten Roboterhersteller erlauben keine abschließende Beurteilung der dynamischen Eigenschaften von konkreten Industrierobotersystemen. Daher werden in diesem Kapitel typische Industrieroboter von zwei unterschiedlichen Herstellern qualitativ und quantitativ untersucht. Die dazu entwickelte Methodik ist auch zur Beurteilung anderer Systeme geeignet, sofern diese im Robotertakt einen Informationsaustausch auf Positionsbasis erlauben, also die Eingabe von Bewegungsbefehlen und die Ausgabe der aktuell erreichten Gelenkwinkel.

3.1 Vorbemerkungen

Kapitel 3 steht in engem Zusammenhang mit Kapitel 6 und Anhang C. Die Testbahnen aus dem Anhang werden in beiden Kapiteln ausgewertet, hier bei Verwendung der Steuerung in einer industriellen Standardkonfiguration, in Kapitel 6 unter Verwendung des vorgeschlagenen adaptiven Systems. Dadurch kann der Beitrag der Arbeit auch quantitativ erfasst werden.

Zur Untersuchung der Bahntreue industrieller Robotersysteme wird als Benchmark ein Satz von standardisierten Referenztrajektorien vorgegeben. Durch explizit festgelegte Mess- und Auswerteverfahren wird eine Bewertung der einzelnen Experimente erreicht, die reproduzierbare, untereinander vergleichbare und

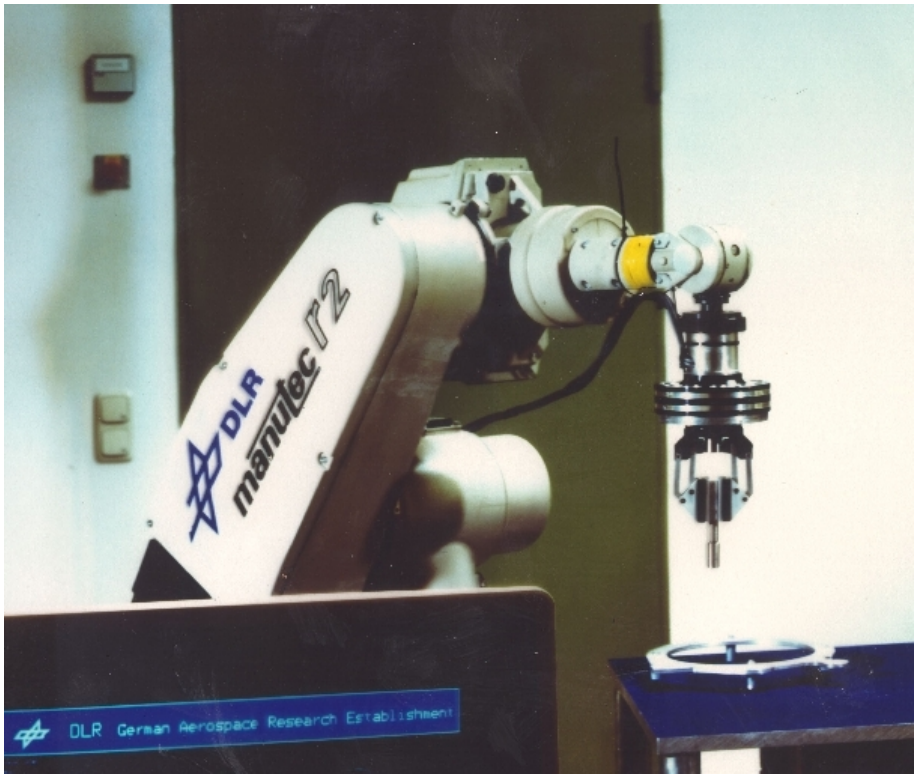


Abbildung 3.1: Testsystem Manutec r2

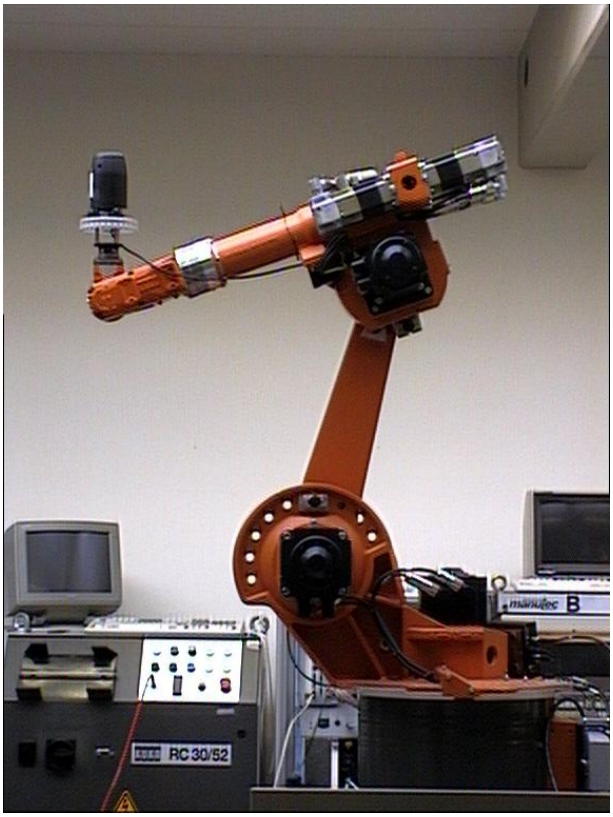


Abbildung 3.2: Testsystem Kuka KR6/1

Abbildung 3.3: Testsystem Kuka KR15/2[139]

auf andere Systeme übertragbare Ergebnisse liefert.

3.1.1 Testsysteme

Die Experimente erfolgen an unterschiedlichen Robotersystemen. Dadurch sollen möglichst allgemein geltende Aussagen gemacht werden. Bei den Robotern handelt es sich um einen Manutec r2 (Abb. 3.1) mit der industriellen Steuerung RCM 3 [225] und um zwei Roboter der Firma Kuka mit der Steuerung KRC1 [137]. Die Roboter von Kuka unterscheiden sich in der Dimensionierung (Maße und Leistung) der Handachsen, wodurch sich zwei verschiedene Gewichtsklassen ergeben. Außerdem sind die Systeme unterschiedlich alt, was sich z. B. in den Getrieben auswirkt. Dagegen ist die kinematische Struktur ähnlich. Die Typenbezeichnungen lauten KR6/1 (Abb. 3.2) und KR15/2 (Abb. 3.3).

Alle Roboter verfügen über industrielle Steuerungen, mit Kaskadenregelung und Sensorschnittstelle ähnlich Abb. 2.2. Sie werden vom Testsystem durch Soll-Gelenkpositionen im Takt von 16 ms (Manutec) bzw. 12 ms (Kuka) angesteuert. Als Ausgang stehen die Ist-Gelenkwinkel (Encoderwerte) und, soweit vorhanden, Sensorwerte im Takt zur Verfügung (siehe Abb. 3.4).

Beim Manutec erfolgt die Kopplung vom Testsystem (Unix) zur Robotersteuerung und zum Kraft- / Momentensensor über einen PC (MS-DOS), der die

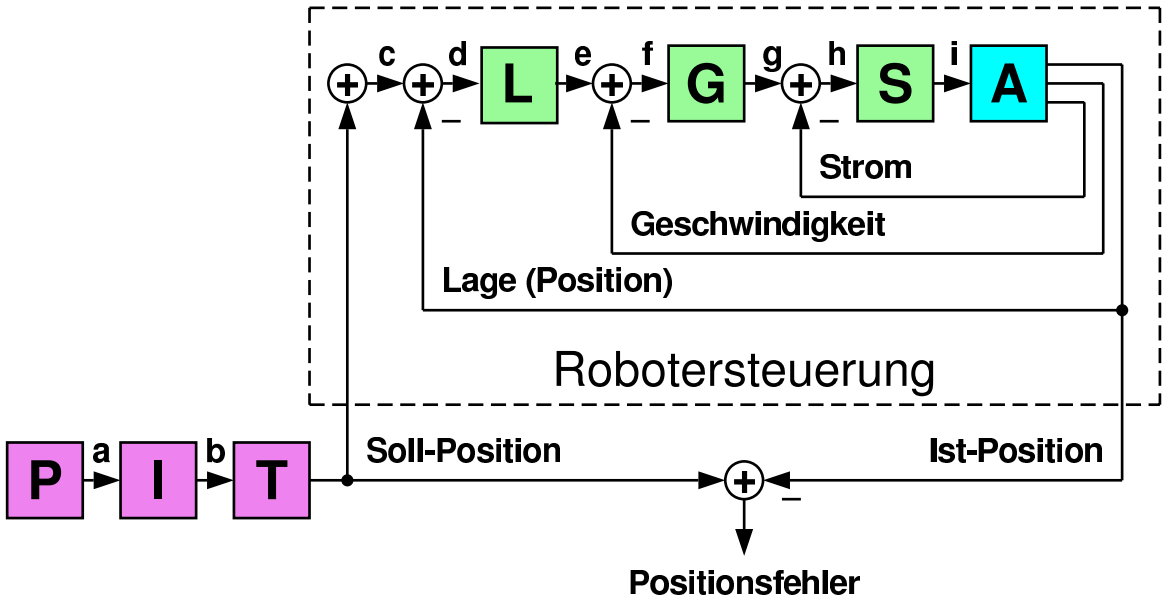


Abbildung 3.4: Struktur zur Bestimmung der Bahngenauigkeit von industriellen Systemen (P = Programmspeicher, I = Interpolation, T = Rückwärtstransformation, L = Lageregler, G = Geschwindigkeitsregler, S = Stromregler, A = Achse, a = Bahnstützpunkte in kartesischen Koordinaten, b = Bahnstützpunkte in Roboterkoordinaten, c = Soll-Position im Takt der Regelung, d = Lagedifferenz, e = Soll-Geschwindigkeit, f = Geschwindigkeitsdifferenz, g = Soll-Strom, h = Stromdifferenz, i = Roboterstellgröße (Strom))

benötigten Daten auf einem Dual Port Memory zur Verfügung stellt und die dort abgelegten Bewegungsbefehle im Takt weiter reicht.

Die Steuerung der Firma Kuka erlaubt die Integration des Testsystems auf dem Steuerungsrechner (VxWorks). Die Berücksichtigung von Sensordaten durch spezielle Hardwarerweiterungen kann auf der gleichen Rechnerplattform erfolgen. Das Bildverarbeitungssystem (LynxOS) ist über eine Socket-Verbindung angeschlossen.

3.1.2 Andere Robotersysteme

Datenblätter von Roboterherstellern wie ABB [1], ADEPT [2], FANUC [80], KUKA [139], REIS [208] oder STÄUBLI [229] geben zur Robotergenauigkeit i. A. nur die Wiederholgenauigkeit an, bzw. die Positioniergenauigkeit im Robotersystem. Diese Werte liegen je nach Gewichtsklasse im Bereich zwischen 0.05 mm und 0.5 mm. Damit ist allerdings keine Angabe über die absolute Genauigkeit (siehe Abschnitt 3.1.3) oder die in dieser Arbeit untersuchten dynamischen Positionsabweichungen gemacht.

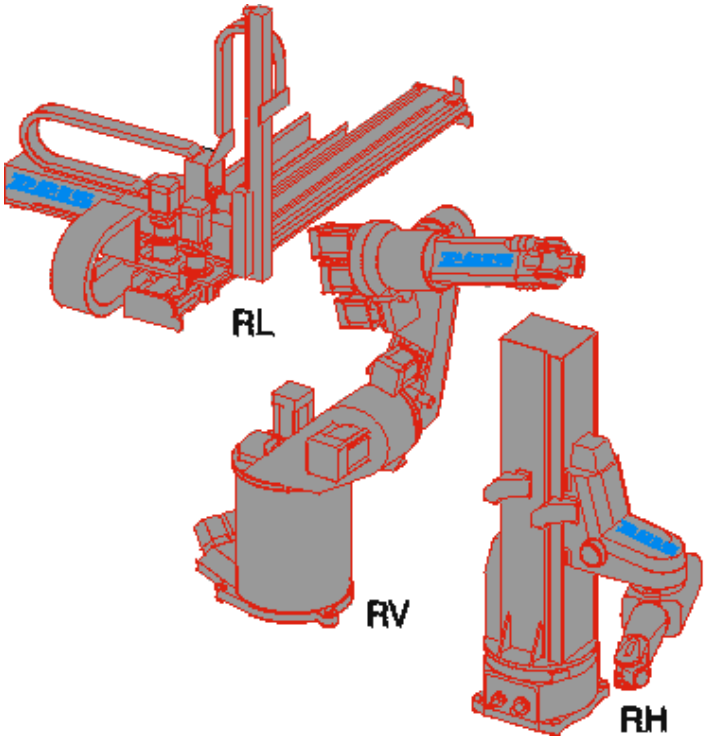


Abbildung 3.5: Bauformen von Robotern nach [208] (RL = Roboter mit Linearachsen, RV = Vertikalknickarmroboter, RH = Horizontalknickarmroboter)

Vergleichende Untersuchungen der Robotergenauigkeit wurden bisher nicht veröffentlicht, da einzelne Experimente nicht miteinander vergleichbar sind. Daher sind definitive Aussagen zur Übertragbarkeit der Ergebnisse nicht möglich.

Es wird angenommen, dass die in dieser Arbeit untersuchten Robotersysteme repäsentativ für vergleichbare Industrieroboterklassen (Vertikalknickarmroboter) sind. Andere Gewichts- und Größenklassen oder Motorleistungen zeigen ein quantitativ anderes Verhalten auf.

Abbildung 3.6: Linearachsenroboter [2]

Abbildung 3.7: Direkt angetriebener Horizontalknickarmroboter (Roboter vom SCARA Typ) [2]

Qualitativ unterschiedliches Verhalten weisen dagegen andere Bauformen (Einteilung siehe Abb. 3.5 oder Carlisle [33]) auf, z. B. Roboter mit Linearachsen (Abb. 3.6) oder Roboter vom Typ SCARA, bei denen die Grundachsen sich senkrecht zur Schwerkraft bewegen (Abb. 3.7). Auch Spezialentwicklungen wie z. B. PORTYS [100], ParaDex [183] oder Leichtbauroboter (Abb. 3.8), wie sie für Weltraumanwendungen oder im Service-Bereich eingesetzt werden, werden im Rahmen dieser Arbeit nicht untersucht. Für die Übertragung der Ergebnisse dieser Arbeit auf solche Roboter sind daher weitere Experimente nötig.

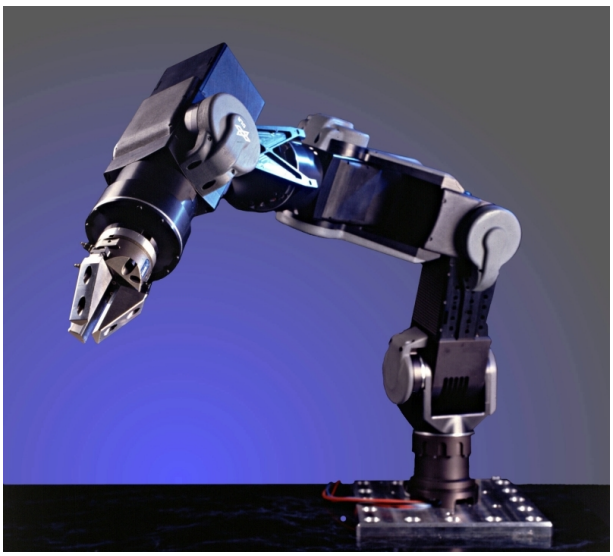


Abbildung 3.8: DLR Leichtbauroboter

Gleiches gilt für getriebelose Roboter, bei denen die Antriebe direkt auf die Roboterarmelemente wirken (z. B. Linearmotoren oder Motoren mit hoher Polpaarzahl), z. B. nach Abb. 3.7.

3.1.3 Absolute Positionsmessung

Die Bewertung der Bahngenauigkeit erfordert zunächst einmal deren messtechnische Erfassung. Bei 6-gelenkigen Robotern ist dies aufgrund der erforderlichen Genauigkeit kein triviales Problem.

In den meisten Fällen wird zur Positionsbestimmung auf die internen Messungen durch die an den Motorwellen messenden Encoder zurückgegriffen. Die so gemessenen Achswinkel ergeben mit der kinematischen Vorwärtstransformation die absolute Position jedoch nur relativ ungenau. Grund sind einerseits eine ungenaue Vermessung oder Referenzierung des Roboters, andererseits verfälschen Elastizitäten in den Getrieben die Armwinkel.

Eine genaue absolute Positionsbestimmung erfordert eine geodätische Vermessung des Roboters in unterschiedlichen Positionen [236]. Dazu werden meist optisch arbeitende Messgeräte (z. B. 3D Laser Tracker, Theodolit, Interferometer oder videogrammetrische Verfahren [22, 159]) angewandt. Aufgrund von vermessenen Positionsdaten kann man die Denavit-Hartenberg-Parameter der kinematischen Transformation so adaptieren, dass der statische Positionsfehler annähernd verschwindet [23]. Zusätzlich werden bei Miyazaki et al. [178] Neuronale Netze zur Kompensation eingesetzt. Bei Drouet et al. [74] wird auch die elastische Verformung des Roboters berücksichtigt. In manchen Fällen [66] ist sogar eine temperaturabhängige Kompensation der statischen Fehler nötig.

Andere Autoren [125] beschreiben Verfahren, um ohne externe Messanordnung die Denavit-Hartenberg-Parameter der kinematischen Transformation zu bestimmen. Dabei werden verschiedene Endeffektorpositionen jeweils mit unterschiedlichen Konfigurationen angefahren. Aus den unterschiedlichen Gelenkstellungen bei gleicher kartesischer Position lassen sich rückwärts die meisten Parameter bestimmen [125]. Ein weiterer Ansatz verwendet eine robotergeführte Kamera, mit der ein Lineal, dessen Lage im Weltsystem bekannt ist, während der Roboterbewegungen beobachtet wird [272]. Dadurch lassen sich sowohl die Kameraparameter als auch die Armlängen und -winkel des Roboters identifizieren.

Einfacher ist eine relative Positionsmessung, also die Verwendung geeigneter Sensorik, durch die der Roboter z. B. den Abstand zu einem Werkstück feststellt. Die relative Positionierung ist nicht nur einfacher, sie hat auch den Vorteil, dass die Aufgabenbeschreibungen meist relativ definiert sind, d. h. durch sie wird auch der Positionsfehler des Werkstücks kompensiert. Außerdem braucht man nicht zwischen kinematischen Modellfehlern und elastischen Verformungen zu unterscheiden [170].

In dieser Arbeit wird insbesondere der dynamische Fehler in der Endeffektorbahn betrachtet, da angenommen wird, dass die statischen Positionsabweichungen aufgrund von Sensoren zum Verschwinden gebracht werden können. Zur Bestimmung des dynamischen Positionsfehlers reicht eine relative Positonsmessung aus. Daher wird im Folgenden nicht weiter auf absolute Positionsmessungen eingegangen.

3.1.4 Positionsmessung bei elastischen Achsen

Ein besonderes Problem sind Elastizitäten zwischen der Werkzeugspitze und dem Ort der Messung, also den Motorwellen (siehe Abb. 3.9), da dadurch neben den statischen auch dynamische Positionsabweichungen auftreten können. Eine externe Messanordnung oder Sensorik kann also auch bei relativ definierten Bahnen nötig sein.

Zur Messung von dynamischen Elastizitäten eignen sich nur Systeme, die für volle Geschwindigkeit ausgelegt sind (Übersicht in [218]). Sie arbeiten meist berührungslos, wie „RODYM 6D“von Krypton [136, 245], „DTRACK“ von

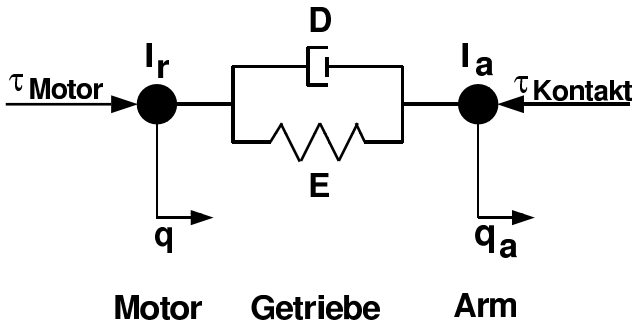


Abbildung 3.9: Mechanisches Modell eines Gelenks (τ = Moment, $\mathbf{I}_r$ = Tragheitsmoment des Rotors, $\mathbf{I}_a$ = Trägheitsmoment des Armes, D = Dämpfungsparameter, E = Elasitizitätsparameter, q = motorseitiger Gelenkwinkel, q_a armseitiger Gelenkwinkel)

ART [14] oder „3SPACE ISOTRAK II“ von Polhemus [200], oder verursachen nur geringe Kräfte, wie „3D CompuGauge“ von Dynalog [75].

Bei einigen Autoren [4, 69, 94, 230, 264] wird nicht die Position gemessen, sondern die Beschleunigung des Endeffektors bzw. die Beschleunigung einzelner Roboterarmelemente. Dies erfordert einen Beobachter zur Bestimmung der Position. Trotzdem ist die Genauigkeit aufgrund von Driften der Integrationskonstanten relativ schlecht, sofern keine hochwertigen Beschleunigungsmesser oder Kreisel verwendet werden.

In Abschnitt 3.4 wird daher ein System zur direkten Positionsmessung verwendet. Dabei zeigt sich, dass die erforderliche Genauigkeit im Bereich von 0,1 - 1 Millimeter nur mit großem Aufwand erreichbar ist, da die Koordinatensysteme von Roboter und Messeinrichtung zumindest statisch sehr genau kalibriert werden müssen.

Außerdem ist die Bewegung mit Messeinrichtung eingeschränkt, da sonst der Messbereich verlassen wird oder spezielle Marken verdeckt werden.

Ein weiterer Nachteil solcher Messsysteme ist, dass die Messungen meist nicht alle 6 Freiheitsgrade berücksichtigen. Bei dem verwendeten System kann z. B. nur die Position in x, y, z Koordinaten bezogen auf das Roboterbasisystem gemessen werden, wobei die Komponente in Bewegungsrichtung wegen mangelnder Synchronisation zwischen Messgerät und Robotersteuerung bei hohen Geschwindigkeiten ungenau ist. Oft lässt sich das Messsystem nicht in die Robotersteuerung integrieren.

Das bringt noch ein weiteres Problem. So liegen die Messwerte online nicht vor. Da nur die Encoderwerte in allen Betriebspunkten im Abtasttakt ausgelesen werden können, werden Positionen zunächst anhand der Encoderwerte gemessen und bewertet. In Abschnitt 3.4 folgt eine Messung mit externem System. Dadurch lässt sich abschätzen, ob die Betrachtung der Encoderwerte ausreicht.

3.1.5 Art der Bahnvorgabe

Bei industriellen Steuerungen sind mehrere Arten der Vorgabe von Teilbahnen üblich [71]:

PTP Punkt zu Punkt Bewegung. Die Bahn zum Zielpunkten ist undefiniert.

LIN Lineare Bewegung. Die Bahn der Werkzeugspitze (Tool Center Point TCP) zum Zielpunkt beschreibt kartesisch eine Gerade. Die Orientierung ändert sich gleichmäßig.

CIRC Zirkulare Bewegung. Die Bahn des TCP zum Zielpunkt beschreibt kartesisch einen Kreis. Außer dem Zielpunkt mit -orientierung wird auch der Kreismittelpunkt vorgegeben.

Andere Bahnformen wie Splines oder polynomielle Darstellungen haben sich dagegen bisher nicht durchgesetzt.

Die Teilbahnen werden normalerweise in 3 Abschnitte geteilt, in denen maximale Beschleunigung, maximale Geschwindigkeit und maximale Verzögerung vorgegeben werden, wobei der zweite Abschnitt bei kurzen Bahnstücken nicht erreicht wird. Neuerdings gibt es auch verbesserte Geschwindigkeitsprofile, die die physikalischen Grenzen wie z. B. die maximalen Gelenkmomente direkt überwachen [79].

An der Schnittstelle von zwei Teilbahnen wird angehalten. Durch Überschleifen kann dies verhindert werden, wodurch allerdings auch die Geometrie der Bahn beeinflusst wird, da nur eine differenzierbare Trajektorie ausführbar ist.

Weitergehende Eingabemöglichkeiten sind meist nicht vorgesehen [137, 225]. So fehlt z. B. die Bahndefinition als Folge von Punkten, durch die ein Polynom gelegt wird.

In dieser Untersuchung wird eine andere Art der Bahnvorgabe verwendet. Eine Bahn ist hier als Folge von kartesischen Positionen und Orientierungen zu den Abtastzeitpunkten definiert. Damit sind sowohl beliebige Bahnen als auch beliebige Geschwindigkeitsprofile vorgebbar. Zur Vermeidung von Mehrdeutigkeiten werden die Bahnen nach der Erzeugung in Achswerte umgerechnet und gespeichert.

Die so definierten Bahnen entsprechen dem, was die Interpolation bei normaler Bahndefinition online berechnet, ggf. mit Sensorkorrektur (siehe Abb. 2.1, 2.2 oder 3.4). Das Argument, dass so definierte Bahnen zu speicherplatzintensiv sind, gilt für die Experimente dieser Arbeit nicht.

3.1.6 Auswahl repräsentativer Kurventypen

Es gibt keine standardisierten Testbahnen zur Beurteilung der dynamischen Genauigkeit. Industrielle Tests beziehen sich entweder auf die Normen [70, 72] oder auf die alte VDI-Richtline [249]. In beiden Fällen werden Geradenstücke vermessen, meist parallel zu den kartesischen Raumachsen, z. B. in Form der Kanten eines Würfels oder Tetraeders.

Die Auswertung beschränkt sich auf die Berechnung verschiedener Kenngrößen wie statischer Genauigkeit oder Überschwingen. Eine allgemeine Beurteilung der Bahnabweichung, wie in Abschnitt 3.1.7 wird nicht vorgenommen.

Eine weitere Benchmark Bahn ist die Schmid'sche Bahn, die eine horizontale Bewegung des TCP bei konstanter Orientierung beschreibt (siehe Abb. 3.10) [70, 215, 219]. Bei dieser Bewegung ist die Form der Bahn gegeben. Größe, Lage im Raum oder Geschwindigkeit werden von jedem Nutzer frei gewählt.

Vorversuche zeigen, dass die größten dynamischen Bahnabweichungen an schnell abgefahrenen Bahnen mit hohen Beschleunigungen auftreten. Die Schmid'sche Bahn erfüllt diese Anforderungen wegen der bezogen auf den Arbeitsraum kurzen Bahnsegmente nur eingeschränkt. Insbesondere eignet sie sich nicht zum Training der in Abschnitt 4.3 vorgeschlagenen adaptiven Vorsteuerung. Auch die anderen Bahnen aus [70] eignen sich mehr zur Messung der statischen Ge-

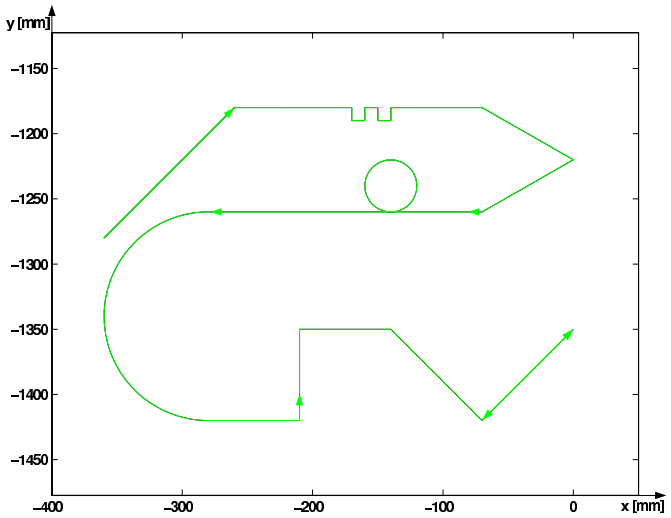


Abbildung 3.10: Schmid'sche Bahn (Die Bahn enthält vor dem letzten Bahnsegment eine Richtungsumkehr)

nauigkeit als der dynnamischen Bahnfehler. Sie werden daher in dieser Arbeit nicht weiter betrachtet.

Das gleiche gilt für Bahnen, die durch eines der Eingabegeräte von Abb. 1.3 bis Abb. 1.6 von einem Bediener vorgegeben werden. Solche Bahnen haben normalerweise geringe Beschleunigungen und Geschwindigkeitkeiten. Außerdem kommt es bei ihnen nicht auf die millimetergenaue Ausführung an, sondern auf eine situationsangepasste Reaktion bezüglich der Sensordaten.

Daher werden neue Testbahnen ausgewählt. In Ermangelung eines geeigneten dynamischen Bahnplanungssystems werden die Testbahnen aus Geradenstücken und Kreissegmenten zusammengestellt, wobei die Geschwindigkeit ortsabhängig vorgebbar ist. Hohe Geschwindigkeiten bei gleichzeitig hohen Beschleunigungen senkrecht zur Bewegungsrichtung lassen sich damit insbesondere bei Kreisbahnen erreichen. Es werden daher verschiedene räumliche Bahnen definiert, die aus Kreisstücken zusammengesetzt sind, ohne dass die Geschwindigkeit an den Übergängen wesentlich reduziert wird. Dabei werden unterschiedliche Konfigurationen (normal, über Kopf; Greifer senkrecht, Greifer waagrecht) beschrieben. Zum Vergleich werden auch zwei Bahnen aus Geradenstücken definiert. Die einzelnen Bahnen sind im Anhang C illustriert. Ein Beispiel zeigt Abb. 3.11.

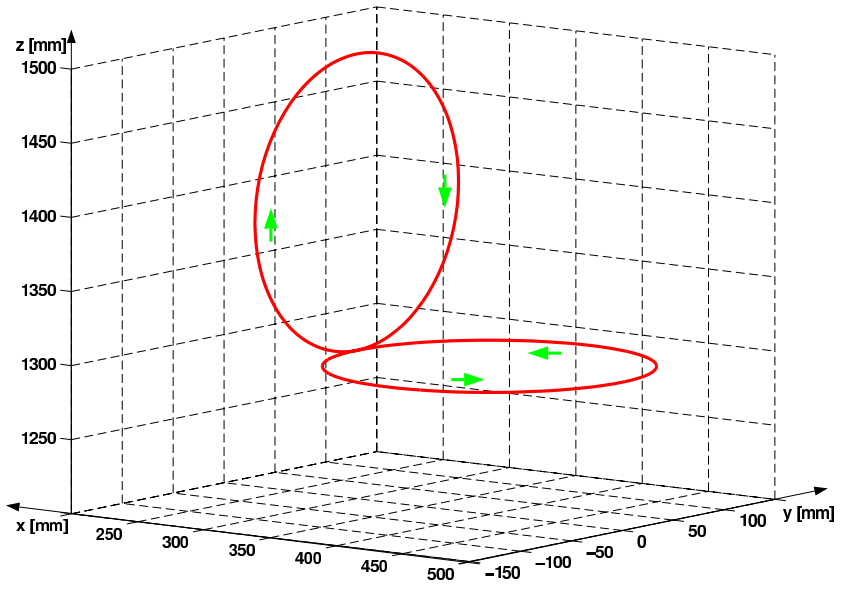


Abbildung 3.11: Typische Testbahn (Bahn 3)

3.1.7 Gütemaße

Zur Beurteilung der Bahngenauigkeit werden zunächst einige Fehlermaße eingeführt. In Abschnitt 5.6 folgen noch modifizierte Gütekriterien für spezielle Anforderungen.

Der Achsfehler (Achspositionsfehler) e_i von Achse i ist definiert als die Differenz zwischen dem (vorgegebenen) Achs-Sollwert (Soll-Gelenkwinkel) w_i und dem Achs-Istwert (Ist-Gelenkwinkel) y_i.

Für Achse i gilt also zum Abtastzeitpunkt k:

$$e_i(k) = w_i(k) - y_i(k) \tag{3.1}$$

Als Gütemaße eignen sich die einzelnen Achsfehler, quadratisch gemittelt über alle Zeitpunkte einer Bahn:

$$J_i = \sqrt{\frac{1}{N} \cdot \sum_{k=1}^{N} e_i^2(k)} \tag{3.2}$$

Von den möglichen Fehlermaßen wird zunächst nur der quadratisch gemittelte Fehler verwendet, da er sich leicht berechnen lässt. Außerdem minimiert das im folgenden beschriebene adaptive Verfahren den quadratischen Fehler. Um den Faktor einer Verbesserung festzustellen, müssen die Wurzeln der quadratischen Fehler verglichen werden. Praktisch interessant wäre neben Mittelwerten auch der maximale Fehler. Er wird zunächst aber nicht explizit ausgewiesen, da er stark von der gewählten Testbahn abhängt. Bei glatten Bewegungen mit betragsmaäßig konstanten Beschleunigungen liegt der maximale Fehler kaum über dem mittleren Fehler, bei Bahnen mit Umkehrpunkten oder anderen Bereichen hoher Beschleunigung liegt er deutlich höher. In erster Näherung kann man annehmen, dass der maximale Fehler um eine Größenordnung über dem mittleren Fehler liegt.

Etwas anschaulicher als das Gütemaß auf Gelenkebene ist ein Ausdruck, der die kartesischen Positionsfehler auswertet. Darunter soll die Differenz zwischen der durch Vorwärtstransformation gebildeten kartesischen Soll-Position $\mathbf{w}_x$ und der entsprechend berechneten kartesischen Ist-Position $\mathbf{y}_x$ verstanden sein. Dabei sind $\mathbf{w}_x$ und $\mathbf{y}_x$ als Vektoren definiert, die die 3-dimensionale Position der

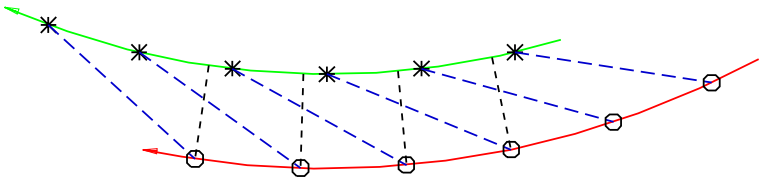


Abbildung 3.12: Vergleich der Positionsfehler (blau gestrichelt) mit den Bahnfehlern (schwarz gepunktet) zwischen Soll-Bahn (grün mit $*$) und Ist-Bahn (rot mit $\circ$) an den Abtastzeitpunkten

Greiferspitze beschreiben. Als weiteres Gütemaß wird also eingeführt:

$$J_x = \sqrt{\frac{1}{N} \cdot \sum_{k=1}^{N} |\mathbf{w}_x(k) - \mathbf{y}_x(k)|^2} \tag{3.3}$$

Dies ist die mittlere kartesische Abweichung der Ist-Bahnpunkte von den korrespondierenden Punkten der Soll-Bahn und kann in Millimetern ausgedrückt werden. Sie enthält aber auch den praktisch unbedenklichen Schleppfehler, d. h. den zeitlichen Versatz zwischen Soll-Bahn und Ist-Bahn, also die Projektion des Positionsfehlers in Bewegungsrichtung.

Die relevante Abweichung der Ist-Bahnpunkte von der Soll-Bahn wird im Gegensatz zum Positionsfehler als Bahnfehler bezeichnet. Der Unterschied ist in Abb. 3.12 dargestellt. Während der Positionsfehler zwei Punkte vergleicht, bildet der Bahnfehler den Abstand eines Punktes von einer räumlichen Bahn.

Mit den Bezeichnungen für den kartesischen Positions- und Bahnfehler sind in dieser Arbeit die translatorischen Abweichungen der Greiferspitze (Positionsbahnfehler) gemeint. Der Orientierungsfehler (Orientierungsbahnfehler) wird dagegen nicht ausgewiesen, da seine absolute Größe hier von untergeordneter Bedeutung ist. Auch die Änderungen des Orientierungsfehlers brauchen nicht gesondert untersucht werden, da man davon ausgehen kann, dass ein Verfahren zur Minimierung der Achsfehler[1] sich sowohl auf die translatorischen Fehler als auch auf die Orientierungsfehler auswirkt.

Ein Bahnfehler lässt sich für die Bewegung einzelner Achsen nicht definieren. Daher bezieht sich der Begriff Bahnfehler im folgenden immer auf den kartesischen Bahnfehler. Er lässt sich geschlossen nur für einfache Bahnen berechnen. So lassen sich z. B. bei analytisch bekannten Kreisbahnen sowohl die Kompo-

[1]Das in dieser Arbeit entwickelte Verfahren zur Erhöhung der Bahngenauigkeit greift auf Achsebene in die Steuerung ein.

nente des kartesischen Positionsfehlers radial zum Kreismittelpunkt als auch die senkrecht zur Bahnebene bestimmen. Diese beiden Komponenten bilden den Bahnfehler.

Bei anderen Bahnen wird in jedem Abtastschritt zunächst der kartesische Positionsfehler berechnet und dann wird die Komponente in Bewegungsrichtung subtrahiert. Vor dieser Berechnung empfiehlt sich eine zeitliche Anpassung von Ist-Bahn und Soll-Bahn, sodass der mittlere Schleppfehler kleiner als einen Abtastschritt wird. Nur dann ist die Berechnung der Komponente in Bewegungsrichtung auch bei starken Krümmungen eindeutig. Die Bewegungsrichtung kann sowohl aus der Ist-Bahn als auch aus der Soll-Bahn bestimmt werden, wobei die Richtung der Ist-Bahn insbesondere bei niedrigen Geschwindigkeiten etwas rauscht. Im Stillstand wird die zuletzt ermittelte Richtung verwendet. Daher wird beim Anhalten nur die Abweichung von der bisherigen Bahn beurteilt, nicht aber ein evtl. auftretendes Überschwingen.

3.2 Messung der Bahngenauigkeit auf vorgegebenen Trajektorien

3.2.1 Freie Bewegungen im Raum

Die ersten Untersuchungen werten Testbahnen aus, die entsprechend Abschnitt 3.1.6 generiert wurden. Tabelle 3.1 vergleicht die verschiedenen Gütemaße bei einer horizontalen und einer vertikalen Kreisbahn mit einem Durchmesser von 100 mm und einer Geschwindigkeit[2] von 375 mm/s bei dem untersuchten Manutec r2 Roboter mit senkrecht nach unten stehendem Greifer. Diese Bahnen werden im folgenden als Bahn 1 und Bahn 2 bezeichnet. Sie sind, wie die übrigen Bahnen, im Anhang C genauer beschrieben.

Man sieht, dass die Achsen den Sollwerten unterschiedlich schnell folgen. Dadurch wird die Wahl des kartesischen Gütemaßes anstelle der Summe der Achsabweichungen unterstützt. In dieses Maß geht der Fehler von Achse 6 wegen des symmetrischen Werkzeugvektors nicht ein. Achse 4 hat in Tabelle 3.1 keinen Fehler, da diese Achse aufgrund der Kinematik des Roboters (siehe Abb. 3.13)

[2]Bei allen Bahnen ist die angegebene Soll-Geschwindigkeit in manchen Bereichen reduziert, damit keine unerlaubt großen Beschleunigungen auftreten. Das betrifft insbesondere Anfang und Ende der Bahnen, die durch konstante Beschleunigungen definiert sind.

Fehlermaß	Kreisbahn 1 (horizontal)	Kreisbahn 2 (vertikal)
Mittl. Positionsfehler von Achse 1	38.730 mrad	34.240 mrad
Mittl. Positionsfehler von Achse 2	50.270 mrad	2.180 mrad
Mittl. Positionsfehler von Achse 3	61.560 mrad	49.640 mrad
Mittl. Positionsfehler von Achse 4	0.660 mrad	0.510 mrad
Mittl. Positionsfehler von Achse 5	18.150 mrad	58.880 mrad
Mittl. Positionsfehler von Achse 6	43.040 mrad	37.370 mrad
Mittl. kartesischer Positionsfehler	23.100 mm	24.060 mm
Komponente senkrecht zur Bahnebene	1.510 mm	3.710 mm
Komponente radial zum Mittelpunkt	1.480 mm	0.940 mm
Mittl. Bahnfehler	2.114 mm	3.827 mm
Mittl. Schleppfehler	23.003 mm	23.754 mm

Tabelle 3.1: Vergleich der verschiedenen Gütemaße an zwei Kreisbahnen (Bahn 1 und Bahn 2)

bei translatorischen Bewegungen mit senkrechtem Greifer nicht bewegt wird. Dadurch kann später der Einfluss von Kopplungen durch andere Achsen untersucht werden.

Bei Betrachtung der Komponenten des kartesischen Positionsfehlers zeigt sich, dass der Schleppfehler gegenüber der Bahnabweichung dominant ist. Dieser Fehler lässt sich bei dynamischen Systemen mit veränderlichen Sollwerten nicht vermeiden.

Abb. 3.14 zeigt Soll- und Ist-Positionen des Manutec r2 bei einer anderen Bahn. Um trotz der engen Beschleunigungsgrenzen des Roboters eine hohe Bahngeschwindigkeit zu erreichen, wurden größere Kreisbahnen gewählt, nämlich ein

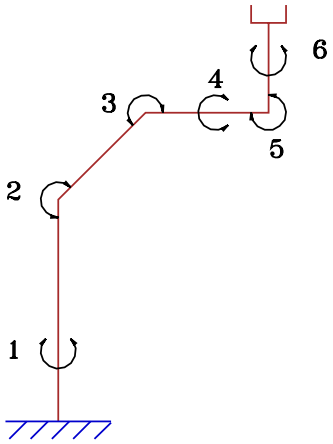


Abbildung 3.13: Schematische Darstellung der Kinematik des Laborroboters Manutec r2 mit den Knickgelenken 2, 3 und 5 und den Drehgelenken 1, 4 und 6

horizontaler und ein vertikaler Kreis mit Durchmesser 200 mm, die nacheinander mit einer Geschwindigkeit von 500 mm/s abgefahren werden (Bahn 3). Dabei zeigt sich ein merklicher Bahnfehler während der vertikalen Bewegung, wogegen der horizontale Kreis kaum Bahnabweichungen aufweist.

Abb. 3.14 zeigt auch, dass die größten Bahnfehler (während des Kreises senk-

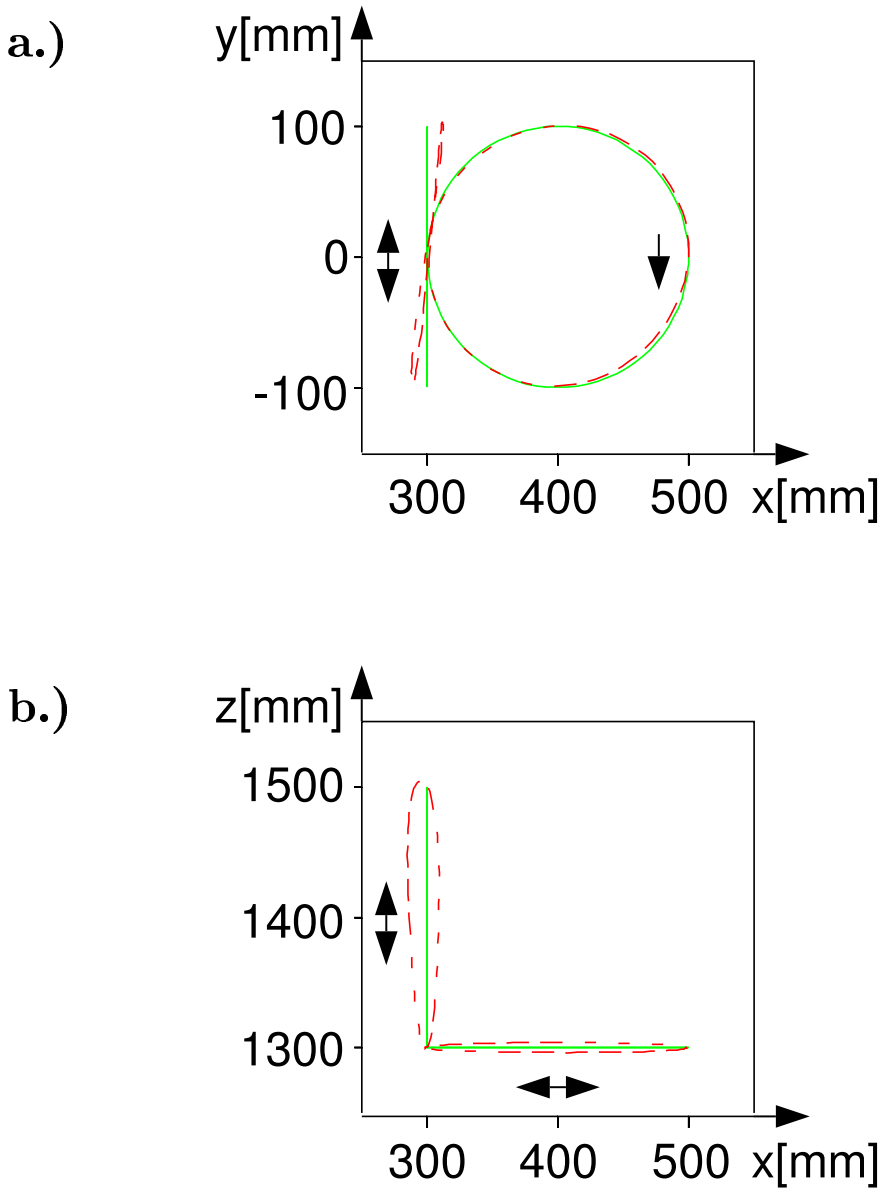


Abbildung 3.14: Soll- und Ist-Werte von Bahn 3 (rot gestrichelt = Ist-Bahn, grün durchgezogen = Soll-Bahn) a.) Projektion von Bahn 3 auf die xy-Ebene, b.) Projektion von Bahn 3 auf die xz-Ebene

Bahn	Max. Geschwindigkeit	Mittl. Positionsfehler
1	375 mm/s	23.091 mm
1	188 mm/s	13.546 mm
2	375 mm/s	24.053 mm
2	188 mm/s	14.126 mm
3	500 mm/s	39.900 mm
3	250 mm/s	21.523 mm
4	375 mm/s	32.074 mm
4	188 mm/s	16.709 mm
5	375 mm/s	27.221 mm
5	188 mm/s	16.218 mm
6	375 mm/s	26.697 mm
6	188 mm/s	15.904 mm

Tabelle 3.2: Mittlere kartesische Positionsfehler des Manutec r2 bei verschiedenen Bahnen (Bahnen siehe Anhang)

recht zur x-Richtung) in einer kartesischen Richtung auftreten, in der zu dem Zeitpunkt überhaupt keine Bewegung stattfindet. Im Achsraum treten sie aber in Richtungen auf, in denen der Roboter sich bewegt. Das heißt, dass die Bewegungen im Achsraum eher entkoppelt sind als bei kartesischer Betrachtung.

Es ist ein weit verbreitetes Vorurteil, dass diese Bahnabweichungen verschwinden, wenn die Achsregler so eingestellt werden, dass alle Achsen das gleiche (lineare) dynamische Verhalten haben. Dies stimmt nur für unbeschleunigte Bewegungen. Zur Erläuterung kann man sich einen x-y-Schreiber vorstellen, bei dem während konstanter Bewegung in x-Richtung plötzlich eine gleichschnelle Bewegung in y-Richtung kommandiert wird. Statt der geforderten Ecke der Bahn wird diese abgerundet, auch wenn bei gleicher Dynamik dann die Soll-Bahn wieder erreicht wird. Solche Bahnabweichungen beobachtet man nicht nur bei unstetigen oder nicht differenzierbaren Bahnen sondern bei allen beschleunigten Bewegungen. Eine Modifikation der Achsregler bringt Bahnabweichungen also nicht in allen Fällen zum Verschwinden.

Tabelle 3.2 zeigt die mittleren kartesischen Positionsfehler des Manutec r2 bei den verwendeten Testbahnen (siehe Anhang C).

Die Bahnen 4 und 5 durchlaufen einen größeren Bereich des Arbeitsraums bei senkrecht nach oben stehendem Greifer. Dadurch kann die Ortsabhängigkeit der Dynamik untersucht werden. Im einen Fall handelt es sich auch um Kreisbahnen, im anderen Fall um Rechtecke. An den Eckpunkten wurde die Soll-Geschwindigkeit reduziert, wie auch an anderen Stellen, an denen sonst die er-

laubten Beschleunigungswerte überschritten würden (vergl. Fußnote auf Seite 57). Auffällig ist die Nähe der Bahnen zur ersten Roboterachse, was bei konstanter kartesischer Geschwindigkeit aufgrund der nahen Singularität eine hohe Achsgeschwindigkeit von Achse 1 und 6 bedeutet. Auf der anderen Seite sind die Gelenke zwei und drei bei der Rechteckbahn fast gestreckt (siehe Abb. C.5), was auch hier aufgrund der nahen Singularität hohe Achsgeschwindigkeiten vorschreibt, diesmal bei den Achsen 2, 3 und 5. Für Achse 1 schwankt die Armträgheit zwischen diesen beiden Extremstellungen fast um den größtmöglichen Wert.

Bahn 6 besteht aus einem horizontalen und einem vertikalen Quadrat, wobei die Quadrate die Kreise von Bahn 3 an den Seitenmitten jeweils berühren. An den Eckpunkten muss die Geschwindigkeit wie bei Bahn 5 reduziert werden, sodass an den Seitenmitten nur eine maximale Geschwindigkeit von 375 mm/s erreichbar ist.

Es zeigt sich, dass der Positionsfehler ungefähr gleich dem in 80 ms zurückgelegten Weg ist, nahezu unabhängig von der Geschwindigkeit und der Art der Bewegung. Stellungsabhängig unterschiedliche Trägheitsmomente und Kopplungen zwischen den Achsen spielen also eine untergeordnete Rolle.

Das kommt daher, dass bei hoch übersetzenden Getrieben die Trägheitsmomente der rotierenden Teile der Motoren gegenüber den an den Roboterarmelementen wirkenden Kräften dominant sind, da Letztere in den dynamischen Gleichungen mit dem Quadrat der Getriebeübersetzung abgeschwächt werden.

Die Bahnabweichung ist davon abhängig, wie sich in der jeweiligen Stellung die Positionsfehler der einzelnen Achsen zueinander verhalten.

Untersuchungen an dem Roboter Kuka KR6/1 bestätigen die Dominanz der mit der Geschwindigkeit annähernd linear wachsenden Schleppfehler, wenngleich die Roboter quantitativ unterschiedlich sind (siehe Tabelle 3.3).

Der Kuka KR6/1 hat bei kartesisch gleichen Bewegungen der Greiferspitze aufgrund der längeren Armelemente kleinere Achsfehler. Kartesisch zeigt er trotz einer größeren Zeitverzögerung einen kleineren Bahnfehler. Der Bahnfehler spiegelt wieder, dass der Roboter aufgrund des großen Schleppfehlers kleinere Kreise beschreibt als kommandiert wurden. Kopplungen der kartesischen Raumrichtungen wie in Abb. 3.14 werden dagegen nicht beobachtet, da die Achsen besser aufeinander abgestimmt sind.

Geschwindigkeit	Manutec r2 500 mm/s	Kuka KR6/1 583 mm/s
Mittl. Positionsfehler von Achse 1	82.816 mrad	70.577 mrad
Mittl. Positionsfehler von Achse 2	61.493 mrad	54.054 mrad
Mittl. Positionsfehler von Achse 3	97.277 mrad	60.721 mrad
Mittl. Positionsfehler von Achse 4	0.260 mrad	0.053 mrad
Mittl. Positionsfehler von Achse 5	70.682 mrad	51.408 mrad
Mittl. Positionsfehler von Achse 6	92.699 mrad	71.746 mrad
Mittl. kartesischer Positionsfehler	39.884 mm	59.619 mm
Komponente senkrecht zur Bahnebene	5.656 mm	0.363 mm
Komponente radial zum Mittelpunkt	2.030 mm	1.789 mm
Mittl. Bahnfehler	6.009 mm	1.825 mm
Mittl. Schleppfehler	39.429 mm	59.591 mm
Zeitverzögerung[3]	93 ms	121 ms

Tabelle 3.3: Vergleich der beiden untersuchten Roboter anhand von Bahn 3 (Beim Kuka KR6/1 ist die Bahn aufgrund der längeren Armelemente nach außen verschoben)

Die Werte von Tabelle 3.3 schwanken zwischen verschiedenen Experimenten um etwa 0.01 mm. Diese hohe Wiederholgenauigkeit bedeutet, dass die Roboter (durch die Kaskadenregelung) eine geringe Empfindlichkeit gegenüber Parameterschwankungen oder Störungen aufweisen. Eine modellgestützte Empfindlichkeitsanalyse nach Frank [86] ist aufgrund der guten experimentellen Ergebnisse nicht erforderlich. Wegen der geringen Modellkenntnis würde sie auch keine genaueren Aussagen erlauben. Eine Aussage über die Empfindlichkeit anderer Robotertypen kann aufgrund der Messungen nicht gemacht werden. Die meist schwere Ausführung von Industrierobotern weist aber allgemein auf eine geringe Empfindlichkeit hin.

Zum Vergleich mit den ausgewählten Testbahnen wird nun die Schmid'sche Bahn nach Abb. 3.10 überprüft. Dabei wird das Geschwindigkeitsprofil gewählt, das die industrielle Steuerung KRC1 standardmäßig erzeugt. Es zeigt sich, dass außer an dem kleinen Kreis praktisch keine Bahnabweichungen auftreten. Das kommt daher, dass an den Eckpunkten jeweils stark abgebremst und ohne Überschleifen sogar angehalten wird. Die Bahngeschwindigkeit erreicht aufgrund der kurzen Abschnitte keine hohen Werte.

Lediglich in dem kleinen Kreis treten hohe Beschleunigungen senkrecht zur Bewegungsrichtung auf. Die Bahnabweichungen beschreiben aber keine durch Zentrifugalkräfte nach außen versetzte Bahn, sondern im Gegenteil einen zu klein ausgeführten Kreis (Abb. 3.15). Das kommt daher, dass der Roboter vom

[3]Die Zeitverzögerung wird am Nulldurchgang von Achse 1 gemessen.

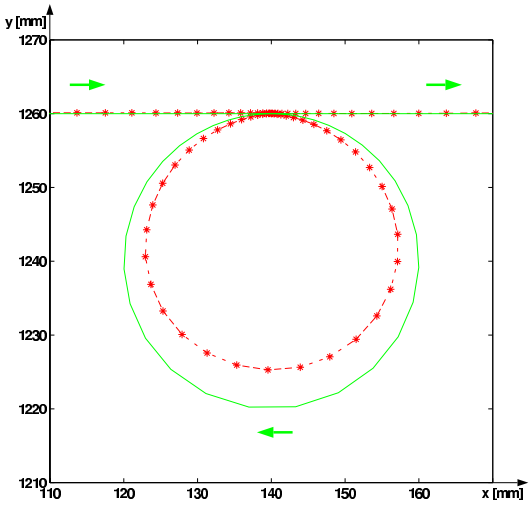


Abbildung 3.15: Soll- und Ist-Werte am kleinen Kreis der Schmid'schen Bahn (rot gestrichelt = Ist-Bahn, grün durchgezogen = Soll-Bahn)

jeweiligen Ist-Wert in Richtung auf den nächsten Sollwert angetrieben wird, wobei der Sollwert durch den Schleppfehler so weit voraus liegt, dass die direkte Verbindung durch den Soll-Kreis verläuft. Die Werkzeugspitze des Roboters wird also in den Kreis hineingezogen. Die maximale Bahnabweichung beträgt dabei etwa 5 mm.

3.2.2 Einfluss von externen Kräften

Die Kaskadenregelung nach Abb. 3.4 sorgt dafür, dass statisch jede kommandierte Position angefahren wird, unabhängig davon, ob eine Gegenkraft besteht. Dies kommt durch einen Integralanteil, den alle Hersteller von Robotersteuerungen in der Kaskadenregelung verwenden. Bei zu starken Gegenkräften, wenn die kommandierte Position aufgrund der Begrenzungen der Motoren nicht erreicht werden kann, schaltet der Roboter ab.

Dynamisch haben Kräfte aber sehr wohl einen Einfluss. Es sind die Änderungen von Kräften, die sich auswirken. Dabei werden in diesem Abschnitt insbesondere die Kontaktkräfte betrachtet, da die intern auf den Arm wirkenden Kräfte (Gewichtskraft, Zentrifugalkräfte und Corioliskräfte) ihre Wirkung nur langsam ändern.

Als Experiment wird mit dem Manutec r2 eine Kreisbahn (Bahn f) abgefahren, die nach Abb. 3.16 einen Kraftkontakt an der Innenkante der Kontur in Abb. 3.1

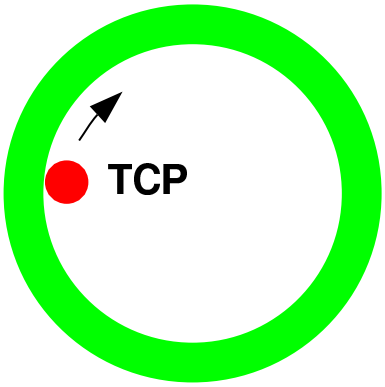


Abbildung 3.16: Anordnung bei Bahn f (roter Punkt = Kontaktstift (TCP) im Robotergreifer an der Startposition, grüner Kreis = kreisringförmige Kontur)

erlaubt. In verschiedenen Testläufen werden unterschiedliche Variationen der Bahn kommandiert, entweder ein Kreis mit einem Radius von 95 mm und einer mittleren Kraft von 3 N oder eine modifizierte Bahn mit einem Radius von 98 mm, was aufgrund des nachgiebigen Sensors etwa 10 N hervorruft. In beiden Fällen wirkt die Kraft auf den im Greifer gehaltenen Stift. Wenn man nun den Greifer öffnet und den Stift entfernt, ergibt sich für die gleichen Bahnen ein anderer Fehler, der der Bewegung ohne Krafteinfluss entspricht. Tabelle 3.4 zeigt das für die beiden Bahnen.

Der Bahnfehler ohne Kontakt ist aufgrund der geringen Geschwindigkeit niedriger als bei den beiden in Tabelle 3.1 aufgeführten Bahnen. Der zusätzliche Fehler durch die Krafteinwirkung ist von der Kraft und der Geschwindigkeit abhängig. So ist in Tabelle 3.1 neben einer Zunahme des Fehlers bei hohen Kräften auch eine Abnahme bei höheren Geschwindigkeiten festzustellen. Das kommt daher, dass die Kontaktkraft dann durch die (auf den Greifer *und* den Arm wirkende) Zentrifugalkraft teilweise kompensiert wird.

Allgemein ist der Krafteinfluss aber gering. Das kommt durch die glatte Bahn,

Bahn	Radius	Geschwindigkeit	Mittl. Bahnfehler ohne Kontakt	Mittl. Bahnfehler mit Kontakt
f	95 mm	31 mm/s	0.213 mm	0.217 mm
f	98 mm	31 mm/s	0.214 mm	0.224 mm
f	95 mm	125 mm/s	0.813 mm	0.805 mm
f	98 mm	125 mm/s	0.871 mm	0.887 mm
1	-	375 mm/s	2.114 mm	-
2	-	375 mm/s	3.827 mm	-

Tabelle 3.4: Einfluss der Kontaktkraft auf den Bahnfehler des Manutec r2

	Mittl. Positionsfehler	
	mit Kontakt	ohne Kontakt
Achse 1	0.082 mrad	0.081 mrad
Achse 2	0.102 mrad	0.101 mrad
Achse 3	0.181 mrad	0.186 mrad
Achse 4	0.893 mrad	0.129 mrad
Achse 5	0.469 mrad	0.309 mrad
Achse 6	0.713 mrad	0.586 mrad
kartesisch	0.305 mm	0.116 mm

Tabelle 3.5: Vergleich der Positionsfehler beim vorgesteuerten Abfahren von Bahn k mit und ohne Kontakt

die selbst bei starkem Andruck und hoher Geschwindigkeit (4. Zeile von Tabelle 3.4) nur Kraftänderungen von etwa 8 N/s aufweist.

Dagegen wird bei der nach Abschnitt 3.3.1 ertasteten Außenkante der Kontur (Bahn k nach Abb. 3.17) eine deutlichere Wirkung festgestellt, da bei dieser Bahn sowohl Betrag als auch Richtung der Kraft sich aufgrund des minimalen Krümmungsradius von 3 mm schneller ändern. Allein die Drehung der Kraftrichtung bewirkt eine Änderung des Kraftvektors um bis zu 130 N/s.

Abb. 3.18 und Tabelle 3.5 beschreiben die Unterschiede an der nach Abschnitt 3.3.1 ertasteten Bahn k, die hier mit 75 mm/s abgefahren wird. Dabei wird eine Positionsvorsteuerung wie in Abschnitt 6.1 verwendet, da sonst die Schleppfehler den Einfluss des Kraftkontakts überdecken.

Tabelle 3.5 zeigt, dass das Verhalten durch die Kontaktkraft insbesondere bei den letzten drei Achsen verschlechternd wird. Kartesisch wirkt sich hauptsächlich Achse 4 aus, erkennbar an den Unterschieden zwischen den y-

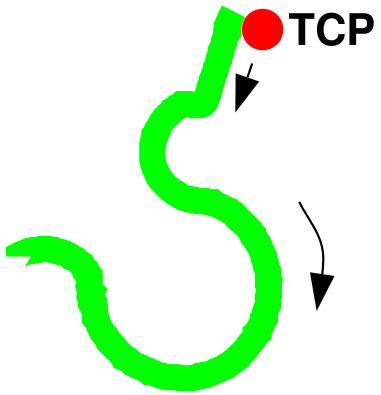
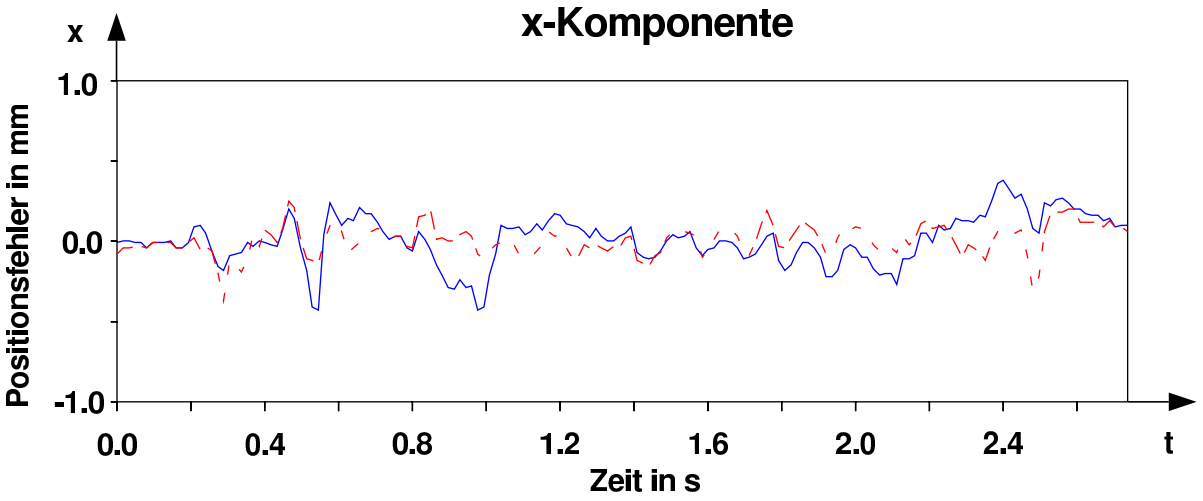
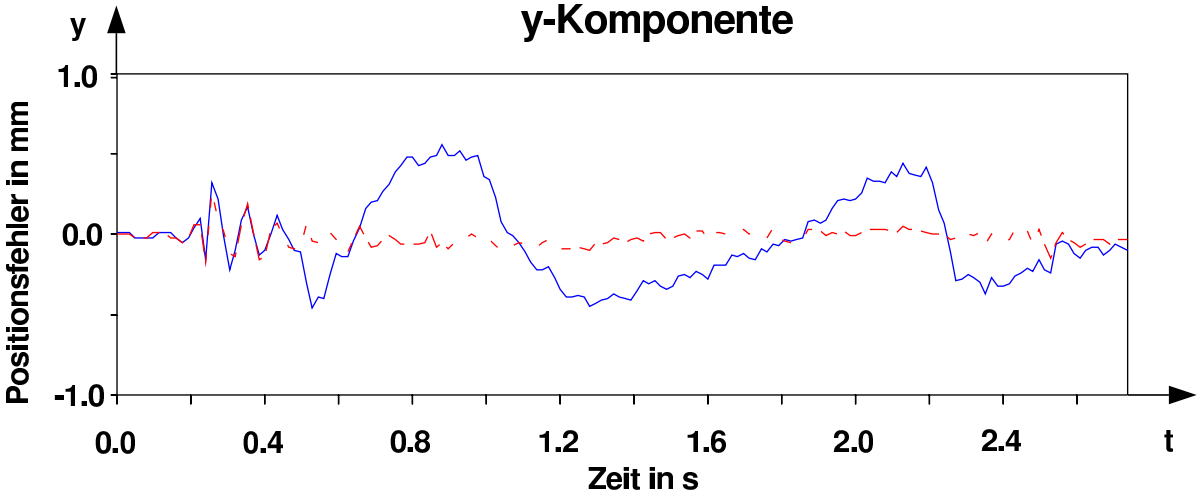


Abbildung 3.17: Ertastete Bahn k (roter Punkt = Kontaktstift (TCP) im Robotergreifer, grüne Linie = Außenkante der Kontur)

a.)



b.)



c.)

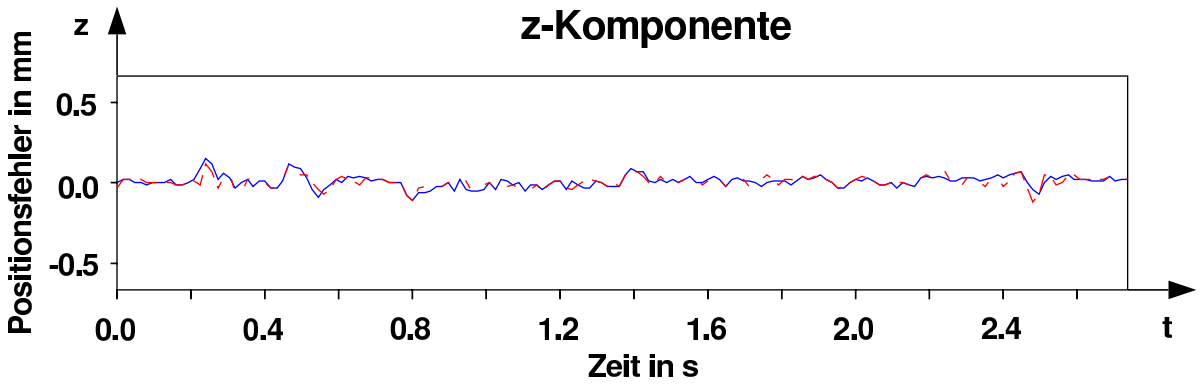


Abbildung 3.18: Vergleich der Positionsfehler bei vorgesteuertem Abfahren von Bahn k mit (blau durchgezogen) und ohne (rot gestrichelt) Stift, also mit und ohne Kraftkontakt
a.) x-Komponente der Positionsfehler, b.) y-Komponente der Positionsfehler, c.) z-Komponente der Positionsfehler

Komponenten beider Bahnen.

3.3 Bahngenauigkeit bei sensorisch modifizierten Bahnen

Bis hierher wurde allgemein von Bahnen ausgegangen, die a priori gegeben sind. Eine hohe Genauigkeit ist in der Praxis aber insbesondere dort erforderlich, wo die Soll-Bahn nahe an Hindernissen vorbeiführt oder einen festen Abstand zu einem Werkstück einhalten soll. Die Position von solchen Hindernissen oder Werkstücken wird nun in der Praxis in vielen Fällen a priori nicht genau bekannt sein, sodass eine sensorgeführte online Bahnplanung eingesetzt werden muss.

3.3.1 Konturverfolgung mit einem Kraft- / Momentensensor

In diesem Abschnitt wird das Beispiel einer durch einen elastischen Kraft- / Momentensensor ertasteten Kontur (siehe Abb. 3.1) beschrieben. Dabei wird die Regelung konventionell, also nach Abschnitt 2.3.1 mit einem trainierten adaptiven Kraftregler aus [152] vorgenommen.

Der Regler unterscheidet nach Abschnitt 2.3.1 eine kraftgeregelte Richtung, die durch die Richtung des jeweils aktuellen Kraftvektors definiert ist, und 5 positionsgeregelte Freiheitsgrade, nämlich die Position in den beiden anderen Raumrichtungen und die drei Orientierungen. Der Sollwert für die Kraft in der kraftgeregelten Richtung beträgt 10 N. In den positionsgeregelten Freiheitsgraden soll nur in einer Komponente eine Bewegung stattfinden. Dazu wird die Bewegungsebene definiert. In den Beispielen dieser Arbeit ist dies die horizontale Ebene. In dieser Ebene liegt auch die kraftgeregelte Richtung, sodass die Bewegungsrichtung, abgesehen vom Vorzeichen, eindeutig ist. Die Soll-Geschwindigkeit wird mit 38 mm/s vorgegeben. Die vertikale Komponente des Kraftvektors wird zur Vermeidung einer durch Messrauschen oder Reibung verursachten Drift nicht ausgewertet.

Das Ertasten der Kontur ergibt bei optimal gelerntem Kraftregler den in Abb. 3.19 gezeigten Kraftverlauf. Die dabei abgefahrene Bahn ist in Abb. 3.17 aufgezeichnet. Man sieht deutlich die Problemstellen, an denen die Soll-Kraft von 10 N nicht eingehalten wird. Dies sind die Orte hoher Krümmung oder

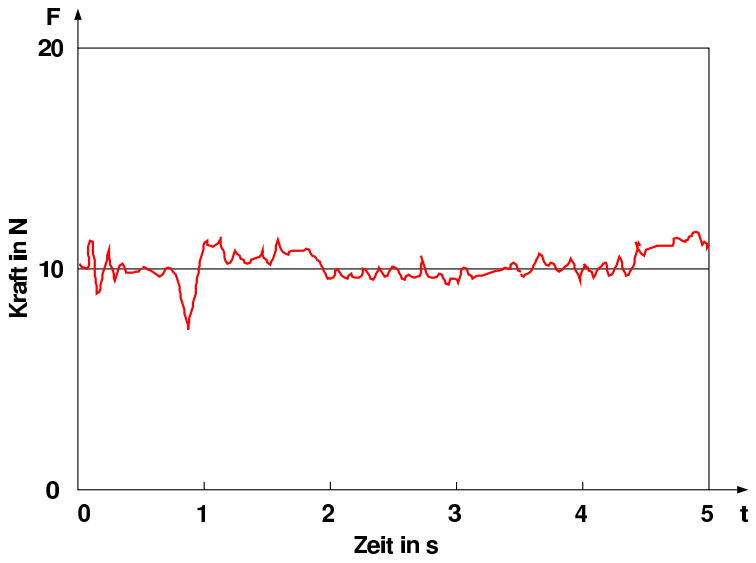


Abbildung 3.19: Kraftfehler bei direkter Kraftregelung mit 38 mm/s

Krümmungsänderung. Der mittlere Kraftfehler beträgt 0.7 N. Er lässt sich nicht reduzieren, da der Roboter die Kontur nur an der jeweils aktuellen Position ertastet und die Totzeit der Signalverarbeitung und die Roboterdynamik keine schnellere Reaktion erlauben. Diese Fehlerschranke kann bei der gewählten Geschwindigkeit auch bei anderen Reglerarchitekturen nicht unterschritten werden. Bei ungünstig gewählten Reglerparametern ergibt sich ein weit höherer Kraftfehler, sofern der Roboter der Kontur überhaupt folgen kann.

Eine Verbesserung des Kraftfehlers kann allerdings durch zusätzliche (prädiktive) Sensoren erreicht werden. So verwenden Baeten et al. in [17, 18] eine Kamera, die den Verlauf der Kontur vorhersagt und eine geeignete Vorsteuerung erlaubt. Voraussetzung dazu ist, dass auch die Orientierung geregelt und vorgesteuert wird, da die Kontur sonst den für die Kamera sichtbaren Bereich verlässt.

3.3.2 Steuerung des Roboters aufgrund von Bildinformationen

Als zweites Experiment zur Bewertung der Bahngenauigkeit an einer durch Sensorsignale online ermittelten Bahn wird das Verfolgen einer durch eine Kamera sensierten Kante betrachtet. Dabei wird ein bildbasiertes Standardverfahren nach Abschnitt 2.3.3 (siehe Abbildung 2.10) eingesetzt.

Im Experiment wird der Roboter mit am Endeffektor montierter Kamera (siehe

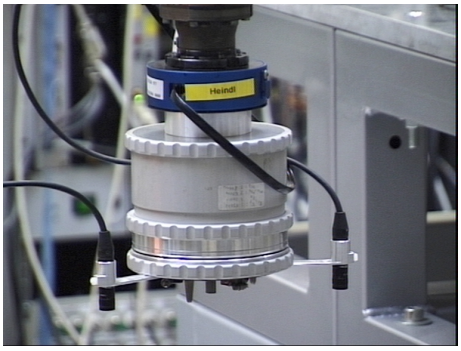


Abbildung 3.20: Endeffektor mit Kraft- / Momentensensor und Stereo Kamerasystem (In diesem Abschnitt wird nur die linke Kamera verwendet)

Abb. 3.20) so programmiert, dass er im Nominalfall entlang einer gut sichtbaren Kante fährt. Während der Bewegung sensierte Abweichungen der Kante vom Nominalfall werden als Bahnabweichungen interpretiert, die online ausgeregelt werden (siehe Abb. 3.22).

Dabei wird die Position der Kante entweder nur aufgezeichnet oder die Position des Endeffektors wird so geregelt, dass die Position im Bild konstant bleibt (Abb. 3.21 und 3.24). Die Regelung erfolgt dabei nach Gleichung (2.3) mit der Merkmalsposition als Sensorsignal $s(k)$. Da die Kamera grob zur Bahn ausgerichtet ist, entspricht $s(k)$ der horizontalen Bildkoordinate der Kante in der mittleren Bildzeile.

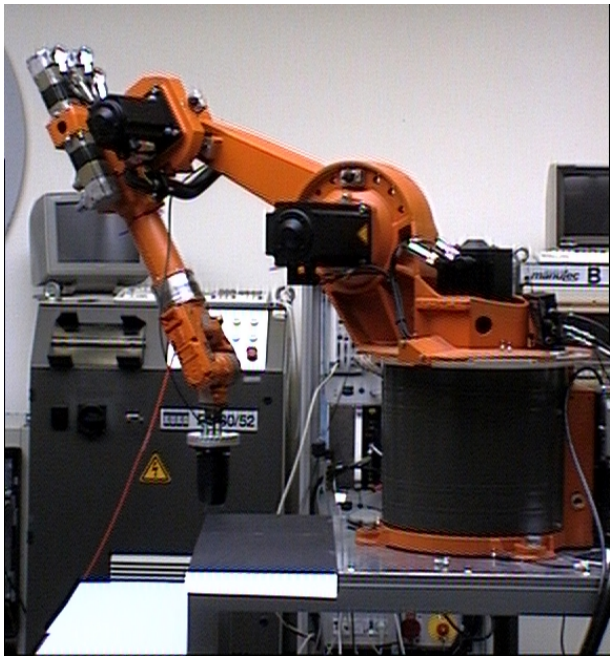


Abbildung 3.21: Roboter bei der visuell geregelten Kantenverfolgung mit in den Endeffektor integrierter Kamera

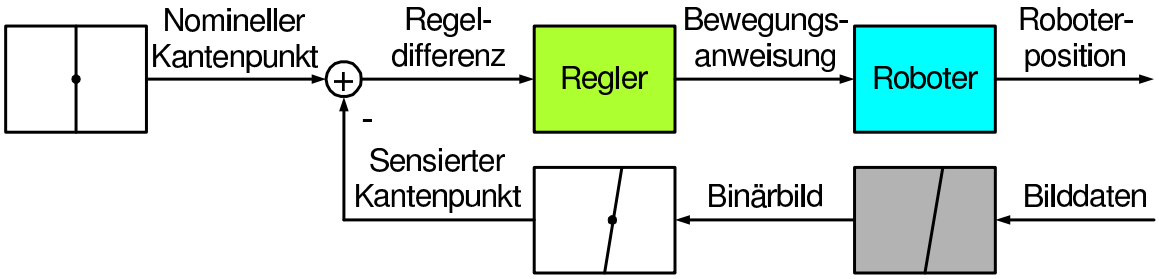


Abbildung 3.22: Korrektur der Roboterbewegung so, dass eine im Bild sensierte Kante eine nominelle Lage einhält

Abb. 3.23 und Abb. 3.25 zeigen ein ähnliches Experiment, bei dem anstelle der geraden Kante aus Abb. 3.21 die Kante nun durch 2 weiße Blätter realisiert ist, die miteinander einen kleinen Winkel bilden.

Bei beiden Experimenten kann die Regelung den Roboter der Kante nachführen. Zwischenzeitlich treten aber, wie bei der Kraftregelung, die unvermeidbaren Regeldifferenzen auf, da die zur Messwertrückführung benutzte Kantenposition selbst die Regelgröße ist.

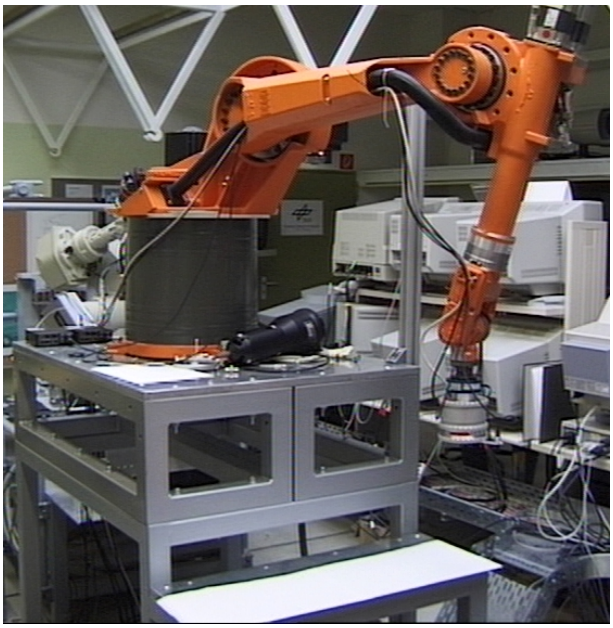


Abbildung 3.23: Roboter bei der visuell geregelten Kantenverfolgung mit Endeffektor aus Abb. 3.20

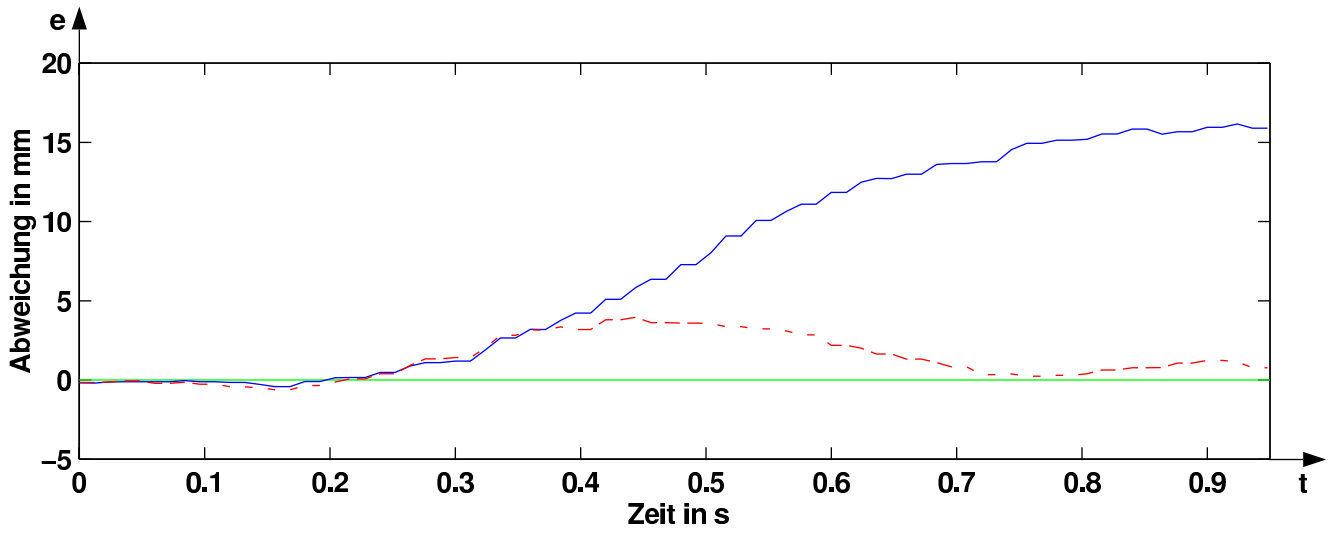


Abbildung 3.24: Abweichung von der geraden Kante (blau durchgezogen = ohne Regelung, rot gestrichelt = mit PD-Regler nachgeführt, grüne Gerade = Soll-Verlauf)

3.4 Verifikation durch externe Messgeräte

In Abschnitt 3.1.4 wurde erläutert, dass in dieser Arbeit die Positionsgenauigkeit zunächst aufgrund der an den Encodern gemessenen Gelenkwinkel beurteilt wird, da diese Werte am besten handhabbar sind. In diesem Abschnitt wird nun untersucht, inwiefern dieser Ansatz zulässig ist, ob die Encoderwerte die tatsächliche Position des Endeffektors also mit genügender Genauigkeit

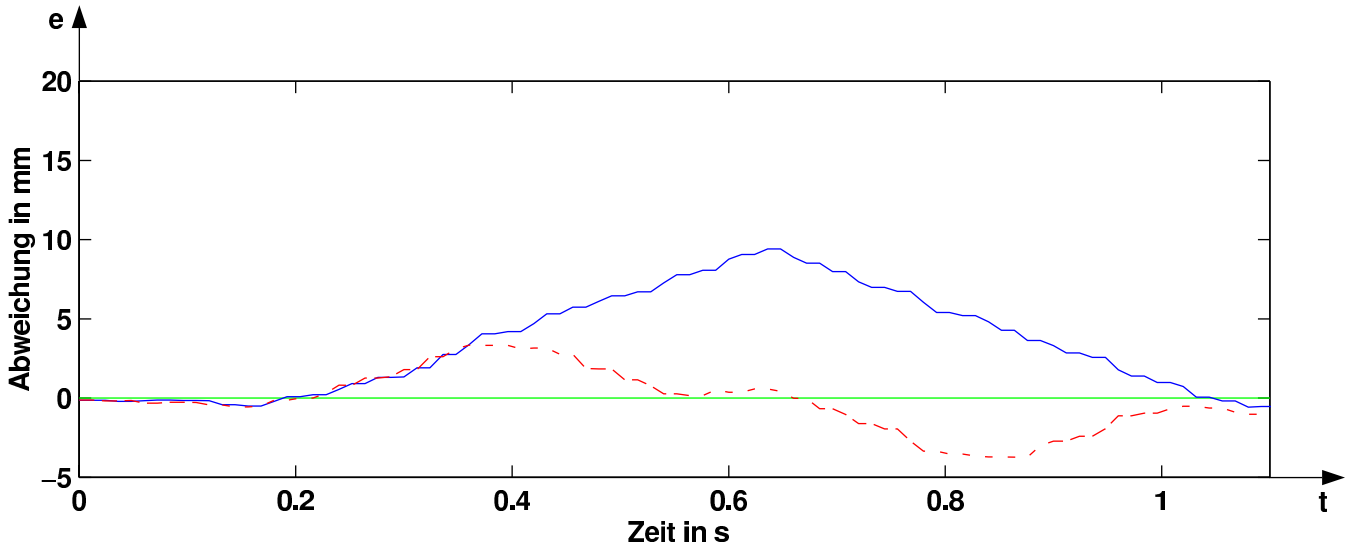


Abbildung 3.25: Abweichung von der Kante mit Knick (blau durchgezogen = ohne Regelung, rot gestrichelt = mit PD-Regler nachgeführt, grüne Gerade = Soll-Verlauf)

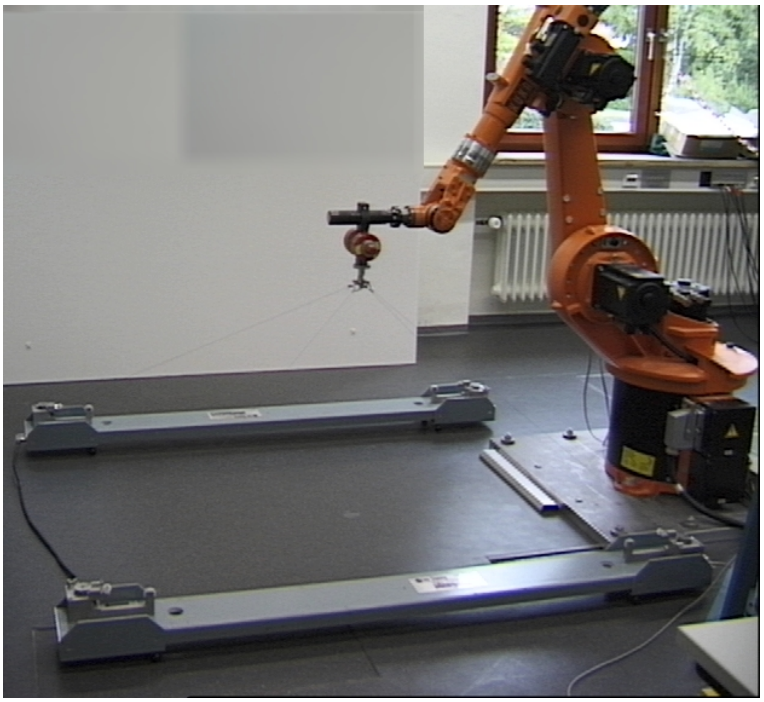


Abbildung 3.26: Roboter mit externer Messanordnung

repräsentieren.

Von den in Abschnitt 3.1.4 aufgeführten Messgeräten wird das „3D CompuGauge“ von Dynalog [75] verwendet (siehe Abb. 3.26 und 3.27). Dieses System hat eine ausreichende Genauigkeit und ist zu dynamischen Messungen geeignet.

Das System besteht aus zwei Messbalken, von deren Enden Seilzüge zum Endeffektor geführt sind (siehe Abb. 3.28). Die Seilzüge sind gespannt und der Abstand vom Endeffektor zu den Messbalken wird gemessen. Die aktuelle Position wird jeweils aus den benötigten Seillängen berechnet. Dazu ist vor jeder Benutzung eine Kalibrierung der Messbalken zueinander und zum Roboterkoordinatensystem nötig.

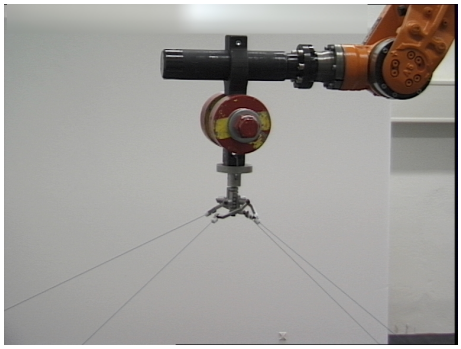


Abbildung 3.27: Endeffektor mit Aufnahme der Seilzüge des Messsystems

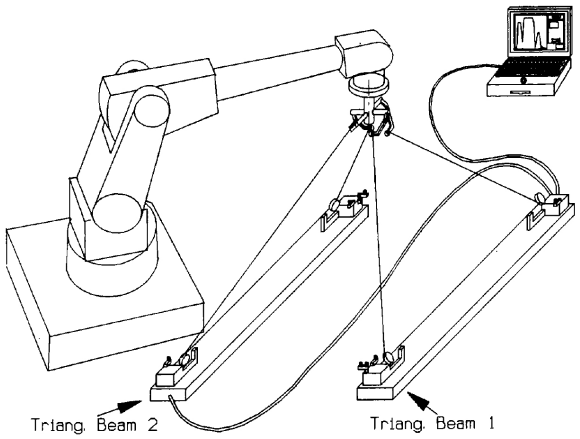


Abbildung 3.28: Prinzip der Messeinrichtung [75]

Es zeigt sich jedoch, dass danach noch statische Unterschiede zwischen den vom Messsystem und von der Robotersteuerung angezeigten Positionen auftreten. Das kommt einerseits durch Ungenauigkeiten bei der Kalibrierung, u. a. aufgrund von Modellannahmen. So wird z. B. angenommen, dass die Messbalken auf einer im Robotersystem horizontalen Ebene stehen. Andererseits machen sich die statischen Ungenauigkeiten des Roboters bemerkbar, insbesondere die schwerkraftbedingten Positionsfehler aufgrund der Nachgiebigkeit der Getriebe.

Aus diesem Grund wird eine zusätzliche Kalibrierung vorgenommen. Und zwar wird die spätere Testtrajektorie zusätzlich noch einmal abgefahren, aber so langsam, dass praktisch keine dynamischen Bahnfehler auftreten. Die bei dieser Trajektorie aufgetretenen Unterschiede zwischen den vom Messsystem und den von der Robotersteuerung ausgegebenen Positionen werden als statische Kalibrierungsfehler bei allen anderen Messungen berücksichtigt. Somit misst das Messsystem nur den dynamischen Fehler.

Die erwarteten dynamischen Unterschiede zwischen den beiden Messungen kommen durch die Nachgiebigkeit der Getriebe (siehe Abb. 3.9), durch die die motorseitigen (antriebsseitigen) und die armseitigen (abtriebsseitigen) Achswinkel sich unterscheiden. Die Getriebeelastizitäten zeigen sich besonders deutlich, wenn das Armträgheitsmoment relativ groß ist. Daher wird bei den Tests in diesem Abschnitt die Nennlastmasse von 6 kg als Endeffektor verwendet (siehe Abb. 3.27).

Zur Wahl der Testbahn gelten die Bemerkungen aus Abschnitt 3.1.6. Dementsprechend sind insbesondere schnelle Bahnen und Bewegungen mit hohen Beschleunigungen geeignet, um Bahnfehler festzustellen. Die Getriebenachgiebig-

Bahnfehler bei Messung durch	Encoder	Dynalog
Teil der Schmid'schen Bahn	0.766 (5.450)	0.626 (5.075)
Teil der Schmid'schen Bahn mit Vorsteuerung	0.200 (1.269)	0.459 (3.225)
Teil der Schmid'schen Bahn an anderem Roboter	0.768 (5.473)	0.584 (4.891)
Bahn 3a	1.834 (2.431)	1.325 (2.200)
Bahn 3a mit Vorsteuerung	0.186 (0.508)	0.642 (1.531)

Tabelle 3.6: Bahnfehler bei unterschiedlicher Messung der Ist-Bahn (Messung der Motorbewegungen durch die Encoder und Berechnung der kartesischen Positionen bzw. direkte Messung der tatsächlichen Armpositionen mit dem Dynalog System. Alle Bahnfehler sind als Mittelwerte in mm angegeben (Maximalwerte in Klammern).)

keit wirkt sich auf den Bahnfehler vor allem bei hohen Beschleunigungen der Lastmasse senkrecht zur Bewegungsrichtung aus, z. B. bei kleinen Kreisen. Aus diesem Grund wird ein Teil der Schmid'schen Bahn (Abb. 3.10) zum Testen ausgewählt. Zum Vergleich wird auch die Bahntreue bei Bahn 3 überprüft, wobei die Bahn in den Messbereich des Messsystems verschoben wird. Zur Unterscheidung wird sie als Bahn 3a bezeichnet (Abb. C.8). Diese Bahn beschreibt eine schnelle Bewegung mit relativ kleinem Armträgheitsmoment für die drei Grundachsen des Roboters.

Beim Vergleich der mit den beiden Systemen gemessenen Positionen muss berücksichtigt werden, dass die Messungen nicht gleichzeitig erfolgten. Die Systeme messen zwar synchron, innerhalb eines Abtastintervalls gibt es aber einen konstanten aber unbekannten zeitlichen Unterschied zwischen den Messzeitpunkten. Daher darf die Positionskomponente in Bewegungsrichtung nicht ausgewertet werden. Andererseits trägt diese Komponente nur zum Schleppfehler, nicht aber zum Bahnfehler bei.

Zur Interpretation der Messungen werden als Kennwerte die Mittel- und Maximalwerte der unterschiedlichen Bahnabweichungen bei ausgewählten Experimenten aufgenommen. Die in Tabelle 3.6 aufgeführten Ergebnisse erlauben insbesondere den Vergleich zwischen unterschiedlichen Robotern gleichen Typs und den Einfluss der Vorsteuerung aus Kapitel 6.

Ohne Vorsteuerung sind die Bahnfehler bei beiden Messverfahren in der gleichen Größenordnung, wobei die der tatsächlichen Bahnabweichungen sogar etwas kleiner sind als die von der Robotersteuerung angezeigten. Die maximalen Unterschiede treten bei der Schmid'schen Bahn dort auf, wo auch die größten Bahnfehler sichtbar sind, nämlich bei dem kleinen Kreis. Auf den ersten Blick scheint es also ausreichend, die an den Encodern gemessenen Bahnfehler zu

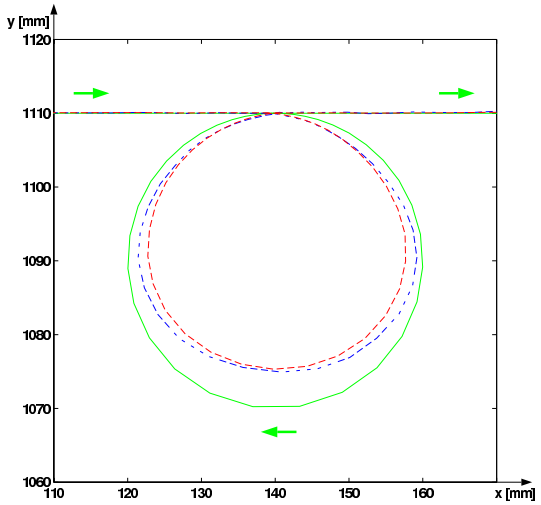


Abbildung 3.29: Vergleich der Bahnen ohne Vorsteuerung bei antriebsseitig (rot gepunktet) und abtriebsseitig (blau gestrichelt) gemessener Position (grün durchgezogen = Soll-Bahn)

reduzieren.

Bei Verwendung der in dieser Arbeit entwickelten Vorsteuerung ergibt sich jedoch ein anderes Bild. So werden die tatsächlichen Bahnfehler nicht in gleicher Weise reduziert wie die dem Verfahren zugrundeliegenden aus den Encoderwerten berechneten Bahnfehler. Abb. 3.30 zeigt, dass zwar die Bahnabweichung aus Abb. 3.29 stark reduziert wird, dass aber dadurch neue Bahnabweichungen entstehen.

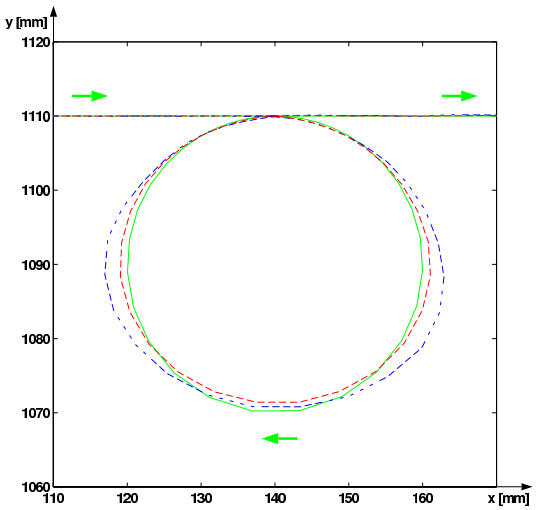


Abbildung 3.30: Vergleich der Bahnen mit Vorsteuerung bei antriebsseitig (rot gepunktet) und abtriebsseitig (blau gestrichelt) gemessener Position (grün durchgezogen = Soll-Bahn)

Ein Verzicht auf externe Messungen ist also nicht möglich. Stattdessen müssen die tatsächlichen Positionsdaten zur Adaption verwendet werden.

Aus Gründen des Messaufwands werden die einmal gemessenen Abweichungen zwischen den tatsächlichen und den von der Robotersteuerung angezeigten Positionen als Kalibrierungsfehler der intern gemessenen Bahn aufgezeichnet und bei weiteren Trajektorien vor der Auswertung und Adaption subtrahiert. Dabei wird angenommen, dass durch Maßnahmen zur Erhöhung der Genauigkeit die Bahn nur so gering verändert wird, dass die Kalibrierungsfehler sich nicht verändern. Abb. 3.29 und Abb. 3.30 zeigen zwar, dass diese Annahme nicht immer zulässig ist. Es ist aber nicht notwendig, dass die tatsächlichen Positionen bei jeder Trajektorie aufgenommen werden. Stattdessen reicht es, wenn die Kalibrierungsfehler während der Adaption überprüft und ggf. neu aufgenommen werden. Abschnitt 6.1.3 bestätigt, dass auch in so schwierigen Fällen wie dem Kreis der Schmid'schen Bahn eine einmalige Anpassung der Kalibrierungsdaten ausreicht.

Die Unterschiede zwischen den beiden Robotern zeigen, welche Genauigkeit man mit vertretbarem Aufwand erreichen kann. So wird in industriellen Anwendungen die externe Messanordnung sicherlich nur einmal vorhanden sein und nicht bei jedem Roboter. Die Getriebefehler werden also nicht individuell berücksichtigt. Gerade dort zeigen sich aber die Unterschiede. So sind in Tabelle 3.6 die an den Encodern gemessenen Bahnen bei unterschiedlichen Robotern aufgrund der Regelung fast identisch, während die tatsächlichen Bahnen sich deutlich unterscheiden. Das kommt durch schwer modellierbare und ggf. abnutzungsabhängige Größen wie Reibung.

Zusammenfassend kann man feststellen, dass zur Adaption eine Kompensation der Getriebefehler sinnvoll ist, sogar wenn nicht mit den realen Daten des Zielssystems gearbeitet werden kann. Bei den meisten Bahnen wird eine einmalige Aufnahme der Messfehler ausreichen.

3.5 Diskussion der Messungen

3.5.1 Genauigkeit der Messungen

Die Genauigkeit der Messungen kann, abgesehen von den Betrachtungen in Abschnitt 3.4, nur durch Vergleich verschiedener Testläufe mit identischen Bewegungsanweisungen beurteilt werden. Dabei zeigt sich eine sehr hohe Wiederholgenauigkeit. Unterschiede betragen meist nur wenige hundertstel Millimeter, die Mittelwerte unterscheiden sich um maximal zehn Mikrometer. Die Trajektorien werden dabei, soweit man dies aufgrund der langsamen Abtastung mit 12 ms erkennen kann, abgesehen von Quantisierungsschritten ungestört gemessen. Die Auflösung der einzelnen Gelenkmesswerte des Manutec r2 ist in Tabelle 3.7 aufgeführt[4]. Lediglich bei Anwendung des in dieser Arbeit vorgeschlagenen Systems sind in einigen Fällen Quantisierungssprünge sichtbar (z. B. Abb. 6.2). Bei den beiden untersuchten Robotern der Firma Kuka ist die Auflösung etwa eine Größenordnung genauer. Damit erlaubt die Messgenauigkeit bei beiden Systemen eine klare Unterscheidung der gemessenen Bahnfehler von Messfehlern. Daher ist eine Filterung der Encoderwerte für die Gelenkwinkel nicht nötig. Die kinematischen Transformationen ergeben entsprechend genaue stetig differenzierbare Bahnen, erfordern also auch keine weiteren Maßnahmen zur Messwertaufbereitung.

Gelenk	Auflösung
1	0.035 mrad
2	0.019 mrad
3	0.078 mrad
4	0.111 mrad
5	0.082 mrad
6	0.101 mrad
kartesisch	0.063 mm

Tabelle 3.7: Auflösung der Messwerte des Manutec r2 (Quantisierung der Encoder)

Die extern gemessenen Positionen sind nicht so genau. Selbst mit der Kalibrierung aus Abschnitt 3.4 muss man mit Messfehlern in der Größenordnung eines zehntel Millimeters rechnen. Dies sind systematische Fehler, die sich durch Filterung nicht reduzieren lassen. Sie kommen durch die (falsche) Annahme, dass der Roboter bei der Kalibrierung der Messanordnung statisch absolut genau ist.

[4]Die Auflösung wird durch die mittlere Differenz zwischen den durch Gleitkommazahlen repräsentierten Kommandos und den aus der internen Festkommadarstellung berechneten so genannten gerechneten Ist-Werten ermittelt.

Aufgrund der Bemerkungen bezüglich der individuellen Unterschiede zwischen verschiedenen Robotern ist die Genauigkeit des Messsystems trotzdem ausreichend. Es wird daher auf eine aufwendigere Kalibrierung oder Aufbereitung der Messwerte verzichtet.

3.5.2 Interpretation der Messungen

Die Untersuchungen in diesem Kapitel zeigen, dass Industrieroboter programmierte Bahnen mit hoher Reproduzierbarkeit abfahren können.

Bei schnellen Bewegungen besteht aber zu jedem Zeitpunkt ein großer systematischer Unterschied zwischen dem zuletzt kommandierten Achs-Sollwert und dem erreichten Ist-Wert. Dieser Schleppfehler entsteht aufgrund der Trägheit des mechanischen Systems und kann daher nur mit hohem Stellaufwand verringert werden.

Im Gegensatz zum Schleppfehler sind die Abweichungen senkrecht zur programmierten Bahn relativ gering. Für den industriellen Einsatz sollen aber gerade diese Abweichungen noch weiter reduziert werden. Da die Bahnabweichungen keinen unmittelbaren physikalischen Hintergrund haben, sondern sich nur mittelbar aus den Positionsfehlern der einzelnen Achsen ergeben, bleibt zur Reduktion der Bahnfehler nur eine Reduktion der Positionsfehler der einzelnen Achsen oder zumindest eine Angleichung der Zeitverzögerungen. Aufgrund der erwähnten mechanischen Trägheit erfordert dies ein prädiktives Vorgehen, vergleichbar mit den Ansätzen aus Abschnitt 2.2.2.

Zusätzlich ist eine Rückführung der tatsächlichen aktuellen Position zur Kompensation von quasi statischen Bahnabweichungen immer dann vorteilhaft, wenn die Umwelt und / oder die Kinematik des Roboters nur unzureichend bekannt sind.

Auch die Kompensation von Getriebefehlern, die durch die Nachgiebigkeit verursacht werden, ist nötig, sofern aus praktischen Überlegungen zunächst die antriebsseitigen Gelenkwinkel verarbeitet werden.

Eine Verbesserung der Achsregelung (Kaskadenregelung) durch adaptive Verfahren erscheint dagegen aufgrund der guten Wiederholgenauigkeit nicht nötig. Auch modellbasierte oder adaptive Ansätze, wie sie in den Abschnitten 2.2.1 oder 2.4.4 aufgeführt sind, versprechen keine grundlegende Verbesserung, da die

limitierenden Größen in der Aktuatorik und Sensorik liegen, nicht aber in der Verknüpfung. So ist einerseits die Bandbreite und das maximale Moment der Motoren begrenzt, andererseits können die Encoder die Bewegung aufgrund der Getriebeelastizitäten nicht genauer beschreiben.

Als wichtigste Anforderung an ein verbessertes System bleibt also die Forderung einer prädiktiven Ansteuerung des Roboters. Dadurch wird im Vergleich zu hoch verstärkenden Reglern die Neigung zu Schwingungen verringert.

3.5.3 Ursachen der Bahnabweichungen

Bahnabweichungen entstehen durch Programmfehler, durch die Robotersteuerung oder durch den Roboter selbst. Programmfehler kommen durch ungenaue Kenntnis der Umgebung z. B. der Werkstückpositionen. Sie lassen sich in den meisten Fällen sensorgestützt kompensieren.

Weitere Bahnfehler werden normalerweise durch die industriellen Steuerungen und Bahnplanungsverfahren erzeugt. Diese Effekte wurden in den Experimenten nur teilweise gemessen, da die industrielle Bahnplanung nicht eingesetzt wurde. Identifiziert wurde z. B. eine Filterung der Kommandos vor der Weitergabe an die Kaskadenregelung [137, 225]. Dadurch wird die Gesamtdynamik deutlich verlangsamt. Solche Maßnahmen werden industriell eingesetzt, um die Anregung von Strukturschwingungen zu vermeiden. Weitere Fehler können durch Rundungsfehler aufgrund von interner Festkommarechnung entstehen, die z. B. zu ungenauen kinematischen Transformationen führt, was bei unseren Testsystemen allerdings nicht beobachtet wurde.

Am Roboter selbst entstehen Bahnfehler durch statische und dynamische Abweichungen. Darüberhinaus hat die Bahn aufgrund von Messfehlern (siehe Abschitt 3.5.1) scheinbar weitere Ungenauigkeiten. Statische Abweichungen ergeben sich durch ungenau produzierte Bauteile, die z. B. einen Winkelfehler zwischen zwei Achsen hervorrufen, durch falsche Referenzierung der Achsen oder durch Elastizität der Roboterarmelemente oder der Getriebe. In Ermangelung einer absolut genauen Messeinrichtung konnten diese Einflüsse nicht quantitativ erfasst werden. Von den möglichen Ursachen für Bahnfehler wurden in diesem Kapitel daher nur die dynamischen Effekte quantitativ untersucht. Dabei können bei den vorhandenen Schnittstellen aber nicht einmal die dynamischen Eigenschaften der Steuerung von denen des Roboters unterschieden werden.

Die Dynamik des Roboters wird durch die Massenträgheit (inkl. Koppelkräften (vergl. Gleichung (2.1))) und die Eigenschaften der Antriebe bestimmt. Dazu gehören neben den zur Verfügung stehenden Motormomenten auch die Massenträgheit der Motoren, die Achselastizitäten und -dämpfung sowie die Reibung.

3.5.4 Schlussfolgerungen

Kapitel 2 und 3 geben den Stand der Möglichkeiten für bahntreue Bewegungen wieder. Dabei zeigt sich, dass die Güte zumindest bei Bewegungen mit maximaler Geschwindigkeit deutlich nachlässt.

Dies liegt zum einen am Bestreben der Industrie, kurze Taktzeiten, also hohe Geschwindigkeiten zuzulassen, was zwangsläufig zu Lasten der Bahngenauigkeit geht. Zum anderen werden industriell bislang nur einfache robuste Regelungsverfahren (Abschnitt 2.1) eingesetzt.

Für verbesserte Systeme ergibt sich aufgrund der Untersuchungen, dass insbesondere die großen dynamischen Verzögerungen und Kopplungen und die daraus resultierenden Bahnabweichungen reduziert werden müssen. Dementsprechend sind Ansätze wie die prädiktive Regelung nach Abschnitt 2.2.2 auf ihre praktische Eignung zu prüfen.

Dagegen ist die statische Wiederholgenauigkeit zufrieden stellend. Die Unterdrückung von stochastischen Störungen ist ausreichend. Systematische Störkräfte haben eine reproduzierbare Wirkung. Eine Verbesserung der unterlagerten Regelung der aktuellen Gelenkpositionen ist also höchstens als Ergänzung einer Vorsteuerung sinnvoll.

Die Getriebeelastizität in den Robotergelenken ist bei den untersuchten Robotertypen unterschiedlich groß. Eine fortschrittliche Steuerung muss daher auch den Fall berücksichtigen, dass elastische Effekte kompensiert werden müssen.

Kapitel 4

Entwurf eines Steuerungsansatzes zur Verbesserung der Bahngenauigkeit

Zur Reduzierung der in Kapitel 3 skizzierten Bahnfehler wird eine adaptive Steuerungsarchitektur entworfen, die verschiedene Ideen aus Kapitel 2 aufgreift und daraus ein neues adaptives Systemkonzept bildet.

Dazu werden bekannte Architekturen untersucht und zu einem neuen dreistufigen Ansatz (Abschnitt 4.2) modifiziert. Neben industriellen Steuerungen, deren Optimierung nicht Gegenstand dieser Arbeit ist, wird eine weitere Regelungs- und Vorsteuerungsebene eingeführt (Abschnitt 4.3), deren Parameter adaptiert werden. Als dritte (oberste) Ebene (Abschnitt 4.4) wird ein Bahnplanungsmodul vorgesehen, das die Daten externer Sensoren integrieren kann.

4.1 Aufgabenstellung

Ziel bei der vorgeschlagenen Steuerung ist eine qualitative und quantitative Verbesserung der Bahngenauigkeit des Roboters. Dazu gehört sowohl das hochgenaue Abfahren vorgegebener Bahnen als auch die sensorische Aufarbeitung und Korrektur einer grob definierten Bahn aufgrund zusätzlicher Messungen.

Anders ausgedrückt, es werden sowohl die dynamischen Bahnabweichungen aufgrund von Trägheiten des Robotersystems untersucht als auch das Problem, die Soll-Bewegung geeignet an die Einsatzumgebung anzupassen.

Das meist durch einen Bediener vorgenommene Teach-in von Roboterbahnen

wird also automatisiert, wodurch insgesamt die Umrüstzeit zwischen zwei Produktionsvorgängen verringert werden kann. Diese Zeit ist bislang sowohl bei Anwendungen in der Großserienfertigung als auch bei Kleinserien ein bedeutender Kostenfaktor.

Neben der Vereinfachung der üblichen mehrstufigen iterativen Vorgehensweise erlaubt das entworfene Verfahren auch eine Optimierung, d. h. die modifizierten Bewegungsanweisungen werden nicht nur schneller generiert, sie erzeugen auch eine höhere Bahngenauigkeit als bei iterativer Bahnkorrektur durch einen Bediener.

Weiterhin kann das System sich durch Verarbeitung von Sensordaten an seine aktuelle Umgebung anpassen, sodass, im Gegensatz zum klassischen Teach-in, die Bahn jeweils unterschiedlich und damit aufgabenangepasst präzise abgefahren wird.

In industriellen Anwendungen sind Systeme erwünscht, die sich, wenn sie einmal adaptiert sind, nicht weiter in ihrem Verhalten modifizieren. Es ist also eine Methode gefordert, die nach kurzer Adaption ohne weitere Modifikation der Regelung oder der Sensorsignalverarbeitung auskommt. Abnutzungsbedingte Nachkorrekturen sind ebenso wie Anpassungen bei Aufgabenänderungen nur auf Initiative des Anlagenbedieners möglich.

Grundsätzlich bieten sich zur Lösung der Aufgabe folgende Teilschritte an:

1. Verbesserung des Regelungsverfahrens unter Nutzung eines inversen Dynamikmodells des betrachteten Roboters
2. Sensorgestützte Korrektur der Bahn (online)
3. Prädiktive Vorsteuerung der gewünschten Bahn

Dabei wird aufgrund der Erkentnisse aus Kapitel 3 die Verbesserung der Regelung auf Basis der Zustandsrückführung allein nicht ausreichen. Problematisch sind dabei die dazu benötigten Rechenleistungen und Schnittstellen, die mit hohen Abtastraten Gelenkmomente generieren. Dazu kommt, dass das inverse Modell nur näherungsweise bekannt ist. Daher wird dieser Ansatz wegen seiner hohen Empfindlichkeit nicht weiter verfolgt.

Die Nutzung von Sensorinformationen erfordert dagegen eine vergleichsweise geringe Rechenleistung. Festzuhalten bleibt jedoch, dass Sensorrückkopplungen

aufgrund des Raum-Zeit-Problems aktuelle Trajektorien nur langsam beeinflussen und darüber hinaus auch verrauscht sind. Eine prädiktive Vorsteuerungskomponente ist also auf jeden Fall sinnvoll. Die folgenden Abschnitte kombinieren daher den Entwurf einer prädiktiven Vorsteuerung mit den Vorteilen zustands-rückführender Regelungsverfahren.

4.2 Ansatz zu einem hierarchischen Steuerungssystem

4.2.1 Voraussetzungen

Voraussetzung zu dem skizzierten Ansatz ist einerseits, dass der Zustand des Roboters vollständig beobachtbar[1] ist und die gewünschte Trajektorie bekannt ist oder sensorisch bestimmt werden kann. Andererseits muss das reale System vollständig steuerbar[2] sein, d. h. man muss online auf den Zustand des Roboters einwirken können.

Da das System breiter anwendbar sein soll, wird von einer bei Industrierobotern üblicherweise vorhandenen Schnittstelle ausgegangen, die es erlaubt, in einem vorgegebenen Takt von z. B. 12 ms Positionskommandos zu verarbeiten und Zustandsmesswerte zu liefern. Dabei wird in Kauf genommen, dass die Messwerte unter Umständen fehlerbehaftet oder verrauscht sind und korrigiert werden müssen (vergl. Abschnitte 4.3.4 und 4.4.3).

4.2.2 Mögliche Architekturen

In der Literatur findet man Ansätze sowohl zu direkten Regelungen als auch zu hierarchisch gegliederten Architekturen zur Regelung und Sensordatenrückführung. Da die Möglichkeiten der direkten Zustandsbeobachtung begrenzt sind (vergl. Abschnitt 2.3), soll hier nur auf die hierarchischen Strukturen eingegangen werden (Übersicht in [93]).

[1] Vollständige Beobachtbarkeit einer Regelstrecke bedeutet, dass der Zustand aus den Ein- und Ausgangswerten eindeutig ermittelt werden kann, z. B. durch einen Zustandsbeobachter. Das setzt nicht voraus, dass alle Zustandsvariablen direkt messbar sind.

[2] Vollständige Zustands-Steuerbarkeit einer Regelstrecke bedeutet, dass eine beliebiger Zustand aufgrund der Stelleingänge erreicht werden kann. Das setzt nicht voraus, dass jede Stellgröße direkt auf nur eine Zustandsvariable wirkt.

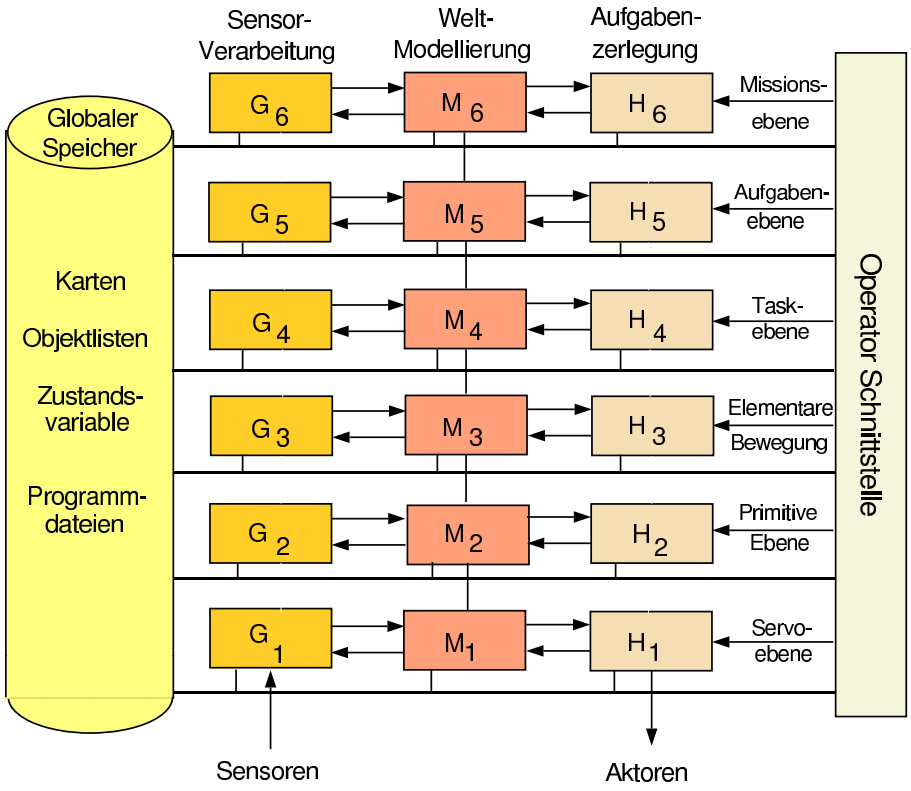


Abbildung 4.1: Hierarchisches Kontrollsystem NASREM [8, 93]

Am bekanntesten ist die NASREM Architektur, die von der NASA entwickelt wurde, um Telerobotiksysteme für Raumflugprojekte zu strukturieren [8, 93]. Insbesondere werden die funktionalen Beziehungen der Multisensordatenverarbeitung beschrieben. Die Architektur besteht aus verschiedenen Ebenen, die jeweils einen Funktionsblock zur Sensorverarbeitung, zur Weltmodellierung und zur Aufgabenzerlegung enthalten, siehe Abb. 4.1.

Die unterste Ebene beschreibt die Servo-Regelung der Roboterachsen, also die Transformation einer kartesischen Bewegung in Gelenkwinkel und die eigentliche Regelung. Sie entspricht bei Industrierobotern einer Kaskadenregelung, enthält aber auch Funktionen zur Koordinatentransformation und eine Feininterpolation.

Die zweite Ebene (Primitive) betrifft die Steuerung der gewünschten Trajektorie auf der Basis einiger vorgegebener Punkte in kartesischen Koordinaten. Dies ist Aufgabe der Interpolation. Dazu gehört auch die Bestimmung eines geeignetes Geschwindigkeitsprofils. Auf der sensorischen Seite werden z. B. die Daten von Kraft- / Momentensensoren eingelesen und verarbeitet.

Die dritte Ebene der elementaren Bewegungen transformiert symbolische Bewe-

gunskommandos in elementare Bewegungen. Dazu gehört die Suche nach einer kollisionsfreien Bahn, sowie die Reaktion auf Informationen der Bildverarbeitung. Die höheren Ebenen der Task- und Aufgabensteuerung sind im Rahmen dieser Arbeit nicht von Bedeutung.

Die NASREM Architektur sieht in der ersten Ebene eine Rückführung der internen Zustandsdaten wie Position oder Geschwindigkeit vor, wie sie bei der betrachteten Roboterschnittstelle in Abb. 3.4 bereits vorliegt. Die Verarbeitung von Sensordaten in kartesischen Koordinaten erfolgt nach NASREM in der zweiten Ebene, die Verarbeitung von Merkmalen aus der Bilddatenverarbeitung ist in der dritten Ebene vorgesehen.

Die Referenzarchitektur der ESA für A&R (Automation und Robotik) wird in [76] als „Functional Reference Model (FRM)“ vorgestellt (vergl. Abb. 4.2). Sie ist in 3 hierarchisch gegliederte Steuerungsebenen unterteilt. Dabei überwacht Ebene A (Aktionsebene) die Ausführung von Elementarfunktionen. Das sind neben einfachen Bewegungsanweisungen (move, approach) auch sensorgestützte Grundoperationen (skills) wie z. B. einfache Montageoperationen (insert). Ebene B (Aufgabenebene) fasst alle Operationen zusammen, die an einem einzelnen Objekt ausgeführt werden. Ebene C (Missionsebene) beschreibt die Aufgabe des Endanwenders.

Die Sicherstellung der hohen Bahntreue von Trajektorien ist bei dieser Architektur auf der untersten Ebene angesiedelt. Die Fehlerbehandlung bei Ausnahmesituationen (nicht-nominelle Rückkopplung) ist nicht Gegenstand dieser Arbeit. Es werden also nur die beiden linken unteren Funktionsblöcke betrachtet, also die nominelle Rückkopplung und die Planung auf Aktionsebene. Betrachtet wird somit die Überwachung der nominellen Prozessdatenverarbeitung, die Sensordatenverarbeitung, die Steuerung und die Regelung, sowie die Aktionszerlegung und die Spezifikation des gewünschten Zustands.

Eine weitere interessante Architektur für Roboter in Raumfahrtanwendungen wurde von Brunner et al. unter dem Namen MARCO (Modular A&R Controller) in [27, 28, 29] vorgestellt (siehe Abb. 4.3). Sie trennt zwischen der offline Programmierung des Roboters und der weitgehend autonomen online Ausführung seiner Aufgabe. Dabei gibt es zwei Hierarchieebenen, die jeweils einen Regelkreis bilden. Auf der oberen Ebene werden implizite Aufgabenbeschreibungen bearbeitet, während auf der unteren Ebene Folgen von explizit sensorbasierten Elementaroperationen ausgeführt werden. Auf der impliziten

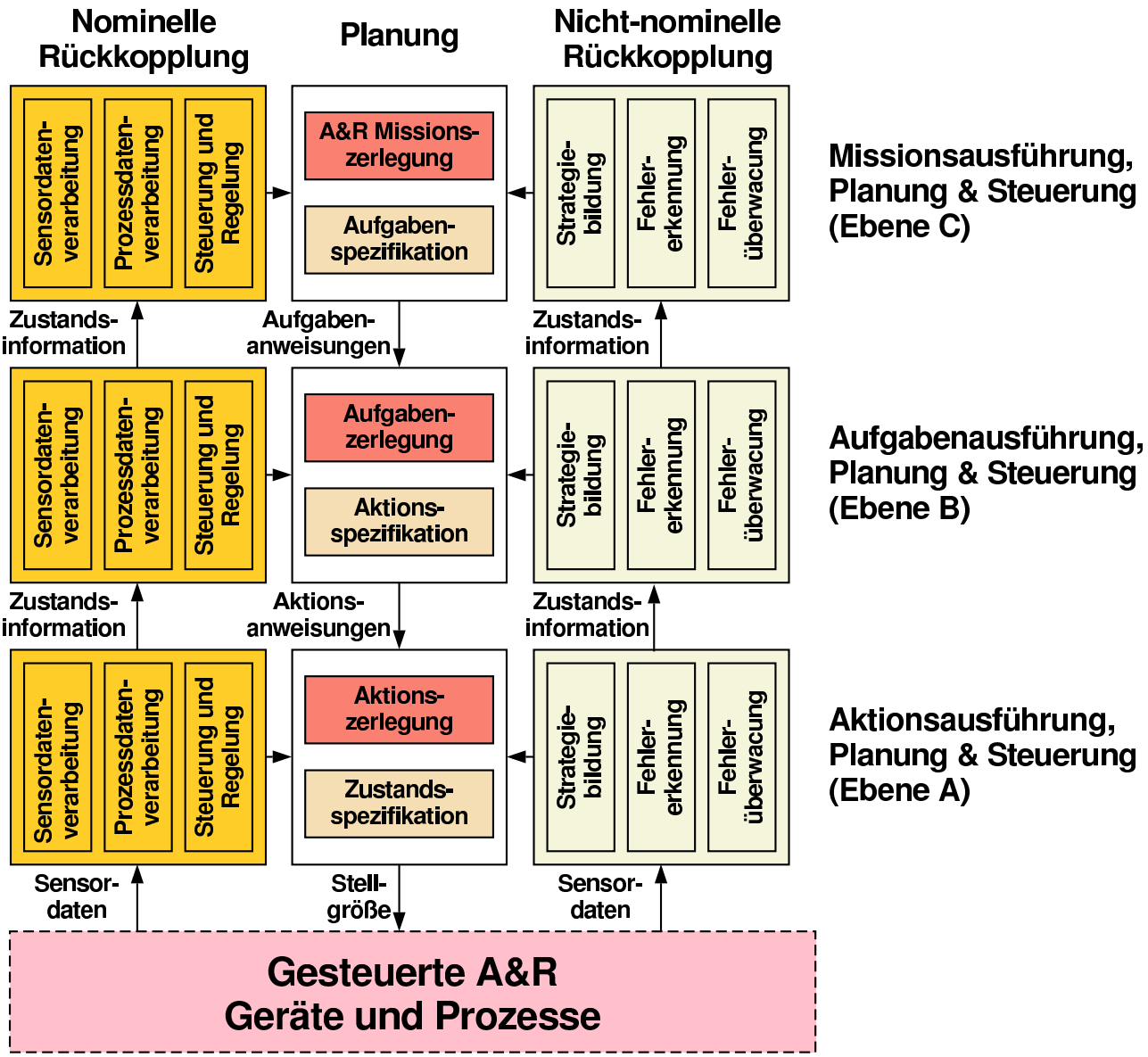


Abbildung 4.2: Hierarchische Architektur FRM [76]

Ebene wird roboterunabhängig beschrieben, *was* getan werden soll. Auf der expliziten Ebene wird fallspezifisch festgelegt, *wie* die Ausführung erfolgen soll.

Man unterscheidet drei Arten von Elementaroperationen: Operationen mit einer Lageregelung, sensorgeführte Operationen und sogenannte Operationen im „shared control mode“. Letzteres bedeutet, dass eine Sensordatenrückführung zur autonomen online Ausführung der Operationen und getrennt eine vorausschauende offline Simulation stattfindet.

Bei Industrierobotern ist der „shared control“ Modus nicht erforderlich, da keine großen Zustandsabweichungen und Signallaufzeiten berücksichtigt werden

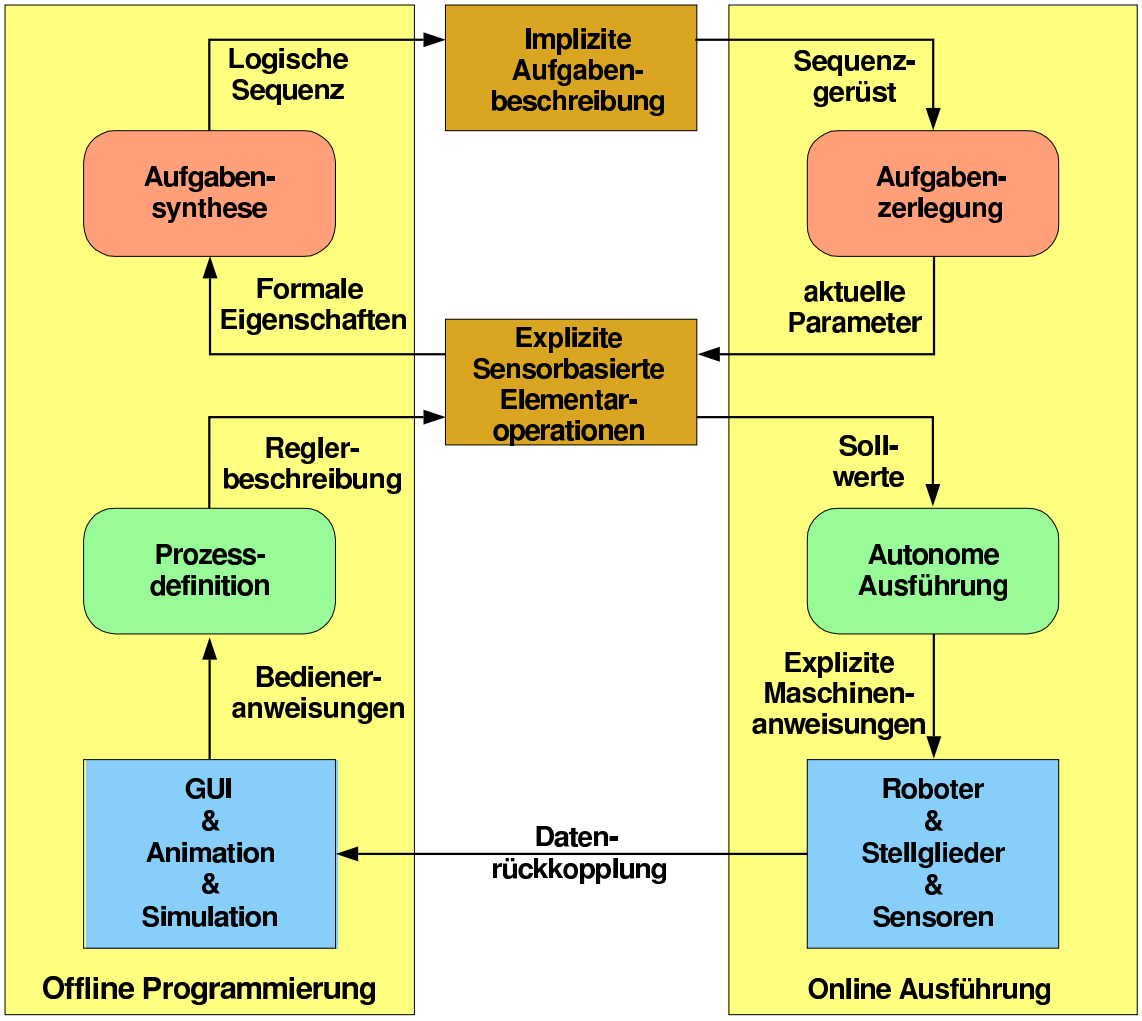


Abbildung 4.3: MARCO Architektur zur Tele-Sensor-Programmierung [27]

müssen. Bei Sensoren mit großer Abtastzeit oder Totzeit ergibt sich allerdings die Notwendigkeit, die Sensorsignale geeignet vorherzusagen um die Bewegungsanweisungen anzupassen.

Die Trennung zwischen implizter (roboterunabhängiger) und expliziter (roboterabhängiger) Ebene ist dagegen für Industrierobotersysteme sinnvoll.

Die in diesem Abschnitt skizzierten Steuerungssystemarchitekturen sind für Systeme mit hoher Autonomie entworfen, die über die Anforderungen an einen Industrieroboter hinausgehen. Zur Sicherstellung der Bahntreue ist daher nur die Betrachtung der unteren Systemebenen sinnvoll.

Eine zusätzliche Hierarchiestufe zur Verarbeitung von externen Sensorsignalen zur Sicherstellung der Bahntreue ist allerdings sinnvoll und bei Industrierobotern durchaus üblich. So werden die Sensorschnittstellen von Industrierobotern

meist niederfrequenter abgetastet als der Lageregelkreis.

4.2.3 Vorschlag zu einem 3-stufigen System

In Anlehnung an die NASREM- und die FRM-Architektur wird eine hierarchische gegliederte Architektur zur Steuerung eines Industrieroboters vorgeschlagen (Abb. 4.4). Die unterste Ebene der Steuerung bildet das industrielle System. Die Optimierung dieser Stufe ist nicht Gegenstand dieser Arbeit.

Die zweite Stufe des hierarchischen Systems ergänzt die industrielle (feedback) Positionsregelung durch eine Vorsteuerung (feedforward), deren Parameter adaptiert werden. Gemeinsam mit den ersten beiden Ebenen der Steuerung bildet der reale Roboter einen *idealen Roboter*. Dieser ist dadurch definiert, dass seine Eingangssignale und Ausgangssignale übereinstimmen, also die Ist-Bewegung y gleich der Soll-Bewegung w ist.

Als oberste Ebene enthält die vorgeschlagene Architektur eine Bahnplanung, die die gewünschte Roboterbewegung spezifiziert. Dabei wird neben einer ana-

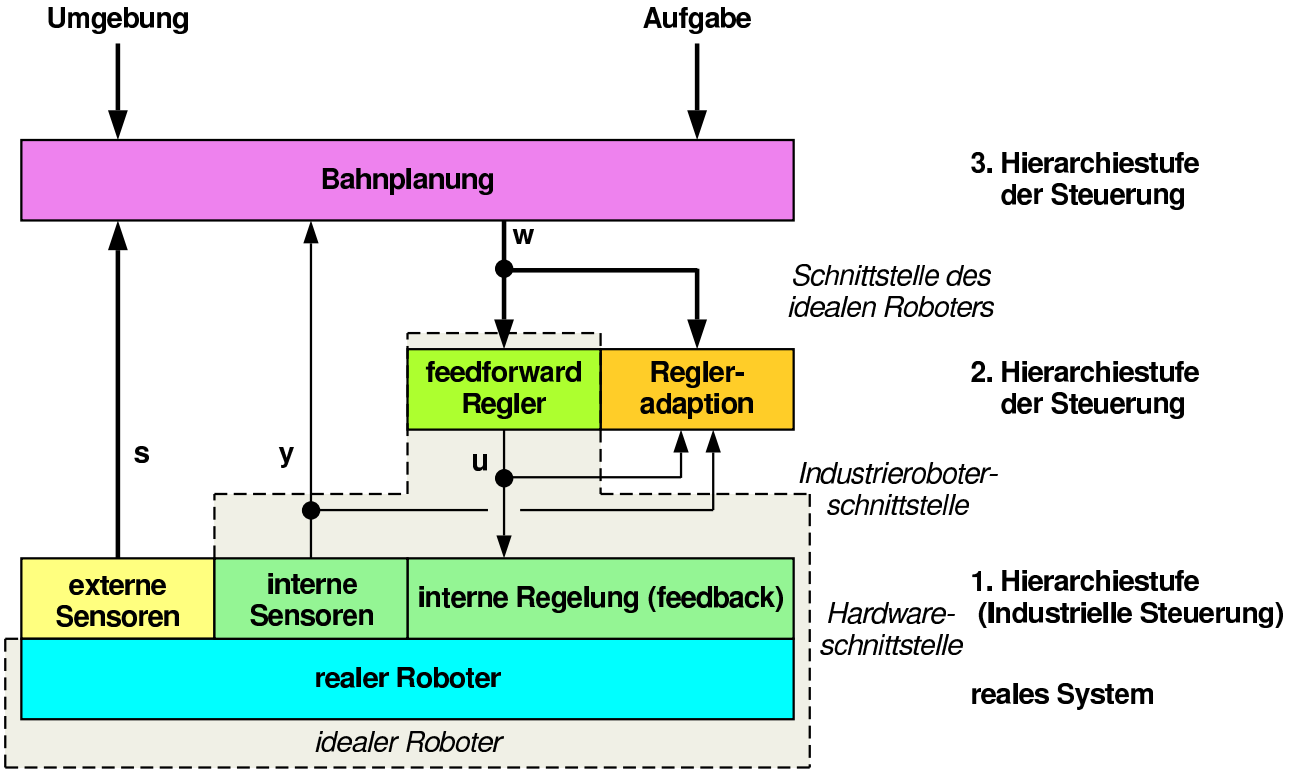


Abbildung 4.4: Hierarchisch gegliederte Darstellung des vorgeschlagenen Systems (w = Sollwerte, u = Stellgröße, y = Messgröße, s = Sensordaten)

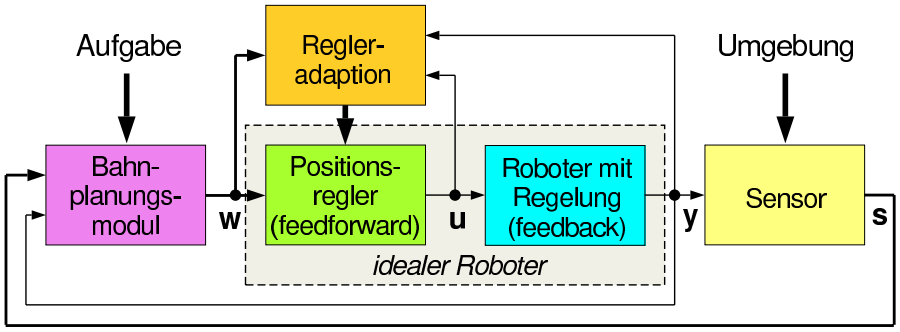


Abbildung 4.5: Struktur der vorgeschlagenen Architektur als Regelkreis (w = Sollwerte, u = Stellgröße, y = Messgröße, s = Sensordaten)

lytisch definierten Trajektorie auch die Information von externen Sensoren genutzt.

Die hierarchische Struktur lässt sich auch als Regelkreis darstellen (Abb. 4.5). Schnittstellen zu dem Regelkreis sind die Ein- und Ausgangssignale des *idealen Roboters*.

Im gestrichelt umrandeten Funktionsblock wird das Verhalten des realen Roboters derart geregelt, dass von außen ein *idealer Roboter* gesehen wird. Dies wird durch eine prädiktive Vorsteuerung erreicht, die so an das reale System adaptiert wird, dass die Bahnfehler minimiert werden. Die Vorsteuerung schätzt die Bewegungskommandos für die industrielle Steuerung, die nötig sind, umd die Soll-Trajektorie auszuführen.

Im äußeren Regelkreis wird aufgabenabhängig die Soll-Bahn des *idealen Roboters* in Form von Soll-Positionen im Takt der Lageregelung bestimmt. Diese Trajektorie wird mit Hilfe des inneren Funktionsblocks der Regelkreisstruktur unmittelbar ausgeführt. Im Idealfall gibt es also keine Regelabweichung.

Um die zweite Stufe der hierarchischen Struktur bzw. den inneren Funktionsblock des Regelkreises, also den *idealen Roboter*, zu realisieren, braucht man neben der Soll-Position zu den aktuellen Zeitpunkten noch weitere Informationen, wie z. B. zukünftige Soll-Positionen oder Ableitungen der aktuellen Soll-Position. Dies bedeutet eine Erweiterung der Schnittstelle. In Übereinstimmung mit den in Abschnitt 2.2.2 beschriebenen Verfahren der prädiktiven Regelung wird die Schnittstelle der Sollwerte so erweitert, dass neben dem aktuellen Vektor jeweils n_w zukünftige Vektoren der Soll-Positionen an die mittlere Ebene übergeben werden. Im Fall der Verarbeitung von Sensorwerten erfordert dies ggf. eine Extrapolation oder Schätzung in der Bahnplanungsebene (vergl. Ab-

schnitt 4.4).

Dabei können sowohl die Ist-Positionen $y(k)$ zum Zeitpunkt k als auch die Soll-Positionen $w(k+i)$ zum Zeitpunkt $k+i$ wahlweise in Achswerten oder in kartesischen Weltkoordinaten angegeben werden. Im Folgenden werden die Roboterzustände in Roboterkoordinaten angegeben, um keine Koordinatentransformation berücksichtigen zu müssen.

In Abschnitt 3.2.2 wurde gezeigt, dass das Roboterverhalten unterschiedlich ist, wenn Kontaktkräfte wirken. Um auf der mittleren Steuerungsebene auch die innerhalb der nächsten n_w Schritten wirkenden Störungen verarbeiten zu können, ist es bei Roboterbewegungen mit Kraftkontakt zur Außenwelt sinnvoll, neben den Positionen auch die einwirkenden Kräfte über die Schnittstelle zur oberen Ebene aufzunehmen. Im Zweifelsfall ermittelt die obere Ebene, ob Kontaktkräfte zu erwarten oder gar erwünscht sind. Konkrete Informationen über bei Soll-Bewegungen auftretende Kräfte können die Regelgüte des Systems verbessern. Diese Kräfte erstrecken sich über den gleichen zeitlichen Horizont wie die Soll-Positionen und müssen im gleichen Bezugskoordinatensystem vorliegen.

Dies ist in Abb. 4.6 dargestellt. Sie unterscheidet sich von Abb. 4.4 nur durch

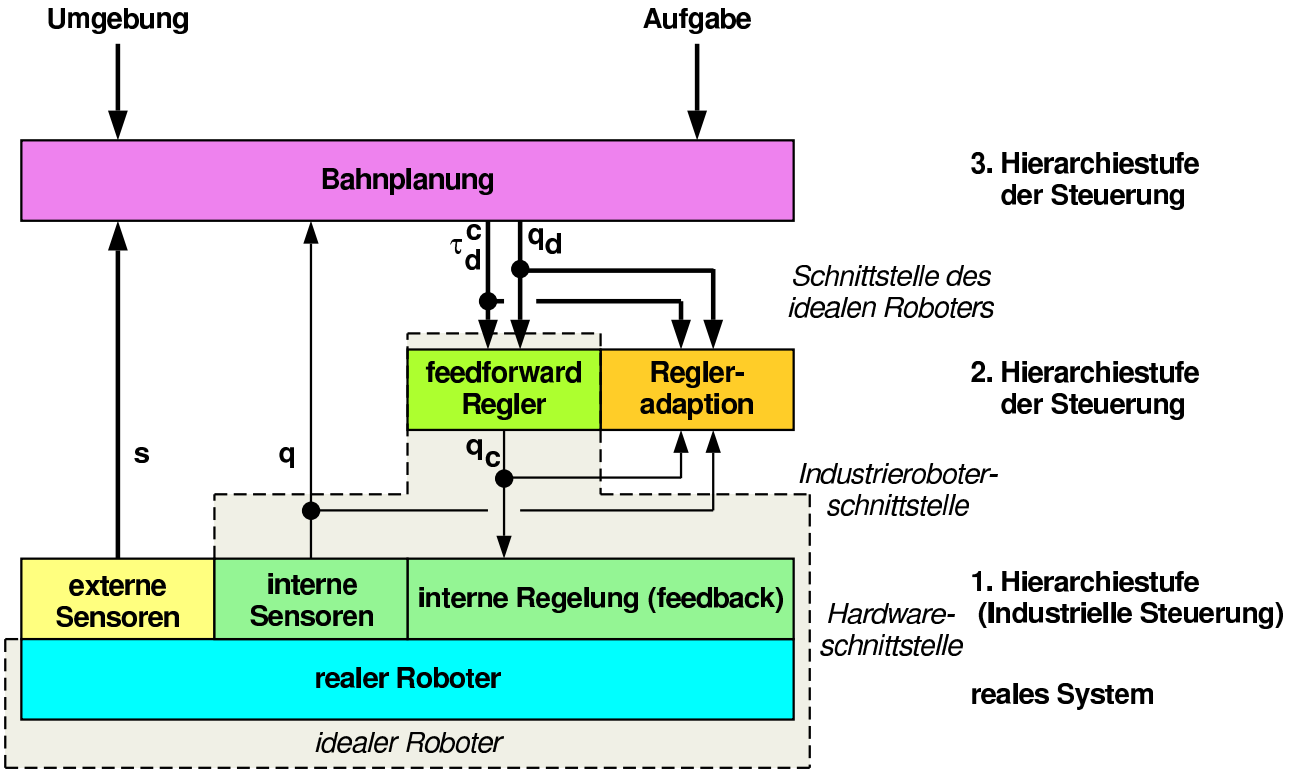


Abbildung 4.6: Hierarchisch gegliederte Struktur mit Berücksichtigung von Kontaktkräften

die Signale. Während in Abb. 4.4 Sollwerte, Stellgrößen und Messgrößen durch die Variablen w, u und y bezeichnet wurden, enthält Abb. 4.6 physikalische Größen. $\mathbf{q}$ sind die Gelenkwinkel des Roboters, $\mathbf{q}_c$ die Bewegungsanweisungen und $\mathbf{q}_d$ die Soll-Gelenkwinkel. τ_d ist der Vektor der Gelenkmomente, der die gewünschte Bewegung ausführt. Dabei wird der Anteil τ_d^c zur Kompensation der Kontaktkräfte und -momente benötigt. Er kann direkt aus den Kontaktkräften berechnet werden (Abschnitt 5.4.3). Anders ausgedrückt werden die Sollwerte $w(k+i)$ durch die 6 Soll-Gelenkwinkel $\mathbf{q}_d(k+i)$ und die 6 Momente $\tau_d^c(k+i)$ gebildet. Entsprechend werden $y(k)$ und $u(k)$ in Abb. 4.4 durch die jeweils 6 Winkel $\mathbf{q}(k)$ und $\mathbf{q}_c(k)$ ersetzt.

4.2.4 Eigenschaften der gewählten Struktur

Die gewählte Struktur hat viele Parallelen zu den in Abschnitt 4.2.2 erläuterten hierarchischen Architekturen, insbesondere zu den unteren Ebenen der Referenzarchitekturen NASREM und FRM. Im Detail gibt es aber Unterschiede, die sich aus den unterschiedlichen Voraussetzungen und Aufgaben für Raumfahrtroboter und industrielle Systeme ergeben.

Es gibt triftige Gründe, die Struktur in der beschriebenen Form festzulegen:

Die Trennung von Regelung und Bahnplanung erlaubt einen getrennten Entwurf der beiden Teilsysteme. Die Regelung kann also durch eine Struktur nach Abschnitt 4.3, durch einen sogenannten „Computed Torque“ Ansatz nach [164] oder andere Verfahren realisiert sein. Wichtig ist nur, dass durch die Regelung eine Näherung des *idealen Roboters* erreicht wird.

Die Realisierung der Bahnplanung ist somit ebenfalls unabhängig von den unteren Ebenen. So können die Daten von Kraft- / Momentensensoren, Entfernungsmessern oder anderen Sensoren verarbeitet werden. Im Gegensatz zu den direkten Regelungsverfahren in den Abschnitten 2.3 und 3.3 können auch mehrere Sensorarten gleichzeitig verwendet werden. Daher wird die Architektur auch als multisensorielle Steuerungsstruktur bezeichnet. Einzige Bedingung ist eine Kalibrierung, sodass aus den Sensordaten die entsprechende Robotersollposition ermittelt werden kann. Im Falle einer Sensordatenfusion bedeutet dies keine Einschränkung, da in jedem Fall eine logische Beschreibung der Sensoren und ihrer Daten existieren muss. Alternativ hierzu kann die Abbildung der Sensordaten auf die Bewegungsanweisungen auch benutzerdefiniert trainiert sein (Skill

Transfer [54]).

Es kann allerdings vorkommen, dass eine geeignete direkte Sensorrückkopplung existiert, ohne dass die Transformation der Sensordaten auf die Roboterkoordinaten genau bekannt ist. Dies ist z. B. dann der Fall, wenn bildbasierte Verfahren der Auswertung von Kamerabildern vorliegen, die Abweichungen von Merkmalen im Bild direkt in Bewegungsanweisungen umsetzen (ohne Bestimmung der räumlichen Positionen der zu den Merkmalen gehörigen Objekte). In dem Fall ist eine strenge multisensorielle Struktur nicht flexibel genug. Stattdessen ist eine ggf. adaptierte direkte Regelung angemessen bzw. die Informationen müssen direkt durchgeschaltet werden.

Weiterhin ungeeignet ist die vorgeschlagene Struktur, wenn die Regelungsebene die erforderliche Genauigkeit nicht erbringen kann. Dies ist bei industriellen Anwendungen in der Regel nicht der Fall. In direkter Interaktion mit dem Menschen, wie z. B. bei mobilen Service-Robotern ist meist eine „weiche“ Regelung gewünscht, also eine Nachgiebigkeit gegenüber äußeren Kräften oder sprunghaften Vorgaben. Für solche Systeme muss die skizzierte multisensorielle Struktur geeignet ausgelegt werden.

Vorteile bringt die vorgeschlagene Struktur insbesondere dann, wenn mehrere unterschiedliche Roboter wahlweise mit unterschiedlicher Sensorik bestückt werden. In solchen Fällen muss die Dynamik für jeden Roboter oder Robotertyp jeweils nur einmal beim Entwurf der beiden unteren Steuerungsebenen berücksichtigt werden. Dies ist Aufgabe des Roboterherstellers. Die Adaption der verschiedenen Elemente des Reglers und der Vorsteuerung erfolgt dann bei der Inbetriebnahme des Roboters.

Die obere Ebene, also die Ebene der Sensordatenverarbeitung, wird von dem Anwender roboterunabhängig entworfen. Bei Wechsel des Robotertyps muss die obere Ebene nicht angepasst werden, solange der Roboter nicht in kritischen Betriebszuständen wie z. B. zu hohen Geschwindigkeiten oder Beschleunigungen arbeitet.

Diese Trennung von aufgabenunabhängiger Kompensation der roboterspezifischen Dynamik und roboterunabhängiger Spezifikation der Aufgabe ist insbesondere bei Anwendungen von Robotern in der Telepräsenz erforderlich.

In vielen Fällen kann es vorkommen, dass der prädiktive Charakter der Schnittstelle nicht zum Tragen kommt. Dies ist dann der Fall, wenn die obere Ebene

zwar die aktuelle Soll-Position, nicht aber die zukünftigen Werte liefern kann. In diesem Fall können die fehlenden Vorgaben für die mittlere Ebene durch Extrapolation der bisherigen Soll-Positionen generiert werden. Eine geeignete Extrapolation ist dabei nicht trivial, da sowohl die Verwendung der aktuellen Soll-Position für alle zukünftigen Schritte, als auch eine lineare Extrapolation oder eine Extrapolation höherer Ordnung keine Garantie für eine ausreichende Repräsentation der gewünschten Bahn geben.

Im Fall von Sensoren, die keine Prädiktion der Soll-Bahn erlauben, kann eine für den speziellen Einsatzfall entworfene direkte Sensorrückführung der vorgeschlagenen Struktur überlegen sein. Das liegt daran, dass beim Entwurf der mittleren Ebene für die vorgeschlagene hierarchische Struktur vorausgesetzt wird, dass die obere Ebene eindeutige Sollwerte erzeugt. Gerade dies ist aber nicht der Fall, wenn zum Zeitpunkt $k+1$ eine andere Soll-Position $\mathbf{q}_d(k+1)$ ermittelt wird als zum Zeitpunkt k für den gleichen Schritt prädiziert wurde ($\mathbf{q}_d(k+1) \neq \hat{\mathbf{q}}_d(k+1) = f(\mathbf{q}_d(k), \mathbf{q}_d(k-1))$). Dies ist in Abschnitt 6.6.2 an einem Beispiel belegt. Stabilität der multisensoriellen Struktur ist aber, zumindest ohne Extrapolation der Sollwerte, immer garantiert, sofern die beiden Ebenen jeweils optimal ausgelegt sind.

Die vorgeschlagene Steuerungsstruktur lässt sich bei üblichen verfügbaren Robotersteuerungen einfach realisieren. Die mittlere Ebene wird als zusätzliches Modul zur vorhandenen Kaskadenregelung (untere Ebene) implementiert (siehe Abb. 4.7). Die obere Ebene erfüllt die Aufgabe des nicht realisierten Moduls zur

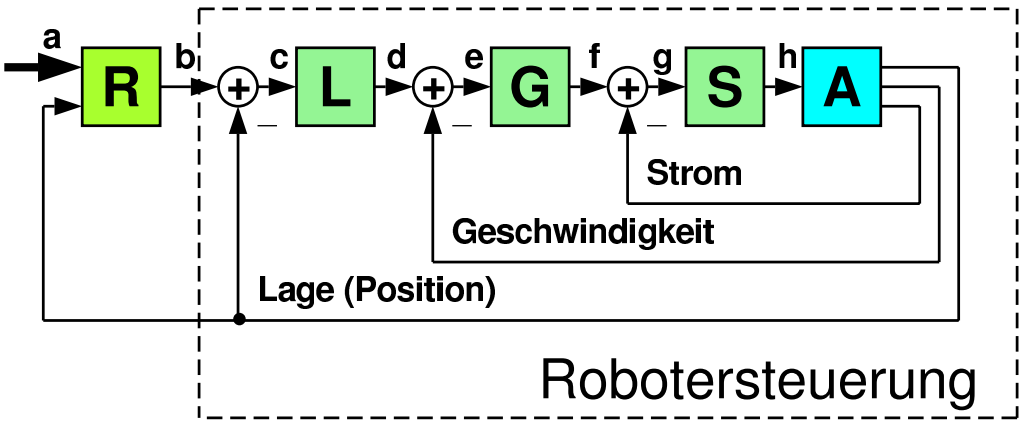


Abbildung 4.7: Struktur des Eingriffs in die Robotersteuerung
(R = externer Regler, L = Lageregler, G = Geschwindigkeitsregler, S = Stromregler, A = Achse, a = Soll-Positionen für mehrere Abtastschritte, b = kommandierte Lage, c = Lagedifferenz, d = Soll-Geschwindigkeit, e = Geschwindigkeitsdifferenz, f = Soll-Strom, g = Stromdifferenz, h = Roboterstellgröße (Strom))

Sensorintegration. Die Schnittstelle des *idealen Roboters* geht allerdings über die in vorhandenen Robotersteuerungen vorgesehenen Zustände hinaus. Deshalb ist eine Integration von beiden Modulen in der Sensorschnittstelle der industriellen Robotersteuerung sinnvoll. Die restlichen Elemente der Steuerung bleiben davon unberührt.

4.3 Vorschlag für eine vorausplanende Steuerung

In diesem Abschnitt wird eine adaptive Methode entwickelt, die übliche Industrieroboter in *ideale Roboter* transformiert.

4.3.1 Annahmen

Das System soll nach der Adaption der Regelung und Vorsteuerung in der Lage sein, auch untrainierte Bahnen mit hoher Genauigkeit abzufahren. Die Aufgabe geht also über das Lernen individueller Trajektorien hinaus, da dabei für jede neue Bahn eine neue Adaption erfolgen muss.

Die Soll-Bahn wird sowohl während der Adaption als auch während ihrer Ausführung als bekannt angenommen, zumindest für die vergangenen und die nächsten zukünftigen Abtastschritte. Gleiches gilt für auftretende Kontaktkräfte.

Weiterhin soll das Verfahren allgemein anwendbar sein, also nicht für die speziellen Eigenschaften eines Robotertyps entworfen werden. Daher wird darauf verzichtet, ein Modell des dynamischen Verhaltens vorzugeben, da solche Modelle schwer zu bestimmen sind [112, 242]. Damit sind modellgestützte Verfahren nach Abschnitt 2.2.1 ausgeschlossen.

Es wird allerdings vorausgesetzt, dass einige qualitative Kenntnisse über das Roboterverhalten vorliegen. Dies sind z. B. die Größenordnungen der erreichbaren Geschwindigkeiten und Bahnabweichungen sowie die Ursache der Störungen.

Die Schnittstelle zum Roboter wird derart gewählt, dass die Hard- und Software handelsüblicher Robotersysteme weitestgehend mitbenutzt werden kann. Dies bedeutet, dass auf einen Eingriff auf der Ebene der Signalprozessoren oder der Leistungselektronik verzichtet wird. Stattdessen wird der bei den meisten

Robotern vorhandene Positionsregelkreis benutzt. Damit liegen zu festen Abtastzeiten die Achspositionen vor, nicht aber deren zeitliche Ableitungen. Die Möglichkeit zur Vorgabe von Positionen in einem festen Zeitraster sei ebenfalls gegeben, sodass die Integration des adaptiven Systems auf der Basis der Positionsschnittstelle nahezu roboterunabhängig implementierbar ist.

Aufgrund der Ansteuerung über die Positionsschnittstelle bedeutet jede Veränderung der Bewegungen des Robotersystems eine Korrektur der vorgegebenen Bahn.

Die Struktur ist in Abb. 4.7 dargestellt. Dabei steht außerhalb der Robotersteuerung nur ein Funktionsblock, dessen Ausgang eine fiktive Position ist, deren Spezifikation das gewünschte Verhalten hervorrufen soll.

Der breite Pfeil als Eingang des externen Reglers deutet an, dass es sich um mehr als ein (vektorielles) Signal wie die Lage oder Geschwindigkeit handelt. Stattdessen ist damit die gesamte Information über die Trajektorie gemeint, soweit sie im Regler benötigt wird. Insbesondere sind es also neben der aktuellen Soll-Position noch mehrere zukünftige Zustandswerte der Soll-Bahn.

In Ermangelung eines online verfügbaren externen Positionsmesssystems wird die betrachtete Bahngenauigkeit, wie in Kapitel 3 gezeigt, zunächst aufgrund der roboterintern gemessenen Gelenkwinkel bestimmt. Die Diskussion der durch diese Voraussetzung resultierenden Fehler folgt in Abschnitt 4.3.4.

Ebenfalls ist die Abtastzeit nicht kritisch. Die beobachteten dominanten Bahnabweichungen sind derart niederfrequent, dass z. B. bei dem Manutec r2 Roboter eine gegenüber dem Takt der Sensorschnittstelle der Steuerung (125 Hz) reduzierte Abtastrate von 62.5 Hz für das adaptive System verwendet werden konnte. Durch die Halbierung der Abtastrate wird der Aufwand des adaptiven Verfahrens überproportional reduziert.

Eine weitere Vereinfachung der vorausplanenden Steuerung ist aufgrund der Aussage möglich, dass der Schleppfehler dem innerhalb einer festen Zeitkonstante zurückgelegten Weg entspricht (Abschnitt 3.2.1). Dadurch ist eine entkoppelte lineare Behandlung in erster Näherung zulässig. Ein nichtlinearer Steuerungsansatz, der ein deutlich längeres Training benötigt, ist allerdings zur Verfeinerung der Steuerung nötig.

Die Vernachlässigbarkeit von Kopplungen konnte bei Experimenten mit einem

Manutec r2 auch am Beispiel geringer Fehler des unbewegten 4. Gelenks (siehe Versuche in Abschnitt 3.2.1) gezeigt werden. Sie gilt allerdings nur bei Regelung im Roboterkonfigurationsraum. So wird in Abb. 3.14 gezeigt, dass kartesische Kopplungen existieren. Bewegung in der y-z-Ebene können Bahnabweichungen in der x-y-Ebene hervorrufen.

In der Literatur (z. B. [91, 163]) wurde vorgeschlagen, zu jedem Abtastzeitpunkt online eine Referenztrajektorie[3]von der aktuellen Ist-Position zur Soll-Bahn zu bilden und anstelle der eigentlichen Soll-Trajektorie die Referenztrajektorie einzuregeln. Dies ist insbesondere dann nötig, wenn große Regeldifferenzen gemessen werden, deren Ausgleich auch unter günstigen Bedingungen mehrere Abtastschritte oder hohe Beschleunigungen erfodert. Aufgrund der hohen Reproduzierbarkeit von Trajektorien ist dieser Ansatz bei Industrierobotern nicht nötig. Lediglich bei online geplanten Bahnen kann eine Zwischentrajektorie notwendig werden, wenn die zu aufeinander folgenden Zeitpunkten berechneten Soll-Bahnpunkte für einen definierten späteren Zeitpunkt sich stark unterscheiden. Für den Fall ist im Bahnplanungsmodul eine Glättung erforderlich.

4.3.2 Anforderungen an Industrierobotersteuerungen

Um das vorgeschlagene System in marktübliche Robotersteuerungen integrieren zu können, müssen einige Bedingungen erfüllt sein.

Eine wichtige Eigenschaft der Robotersteuerung ist sicherlich das Vorhandensein einer Sensorschnittstelle. Der Robotersteuerung sollten also zusätzlich zu den interpolierten Soll-Bahnpunkten noch Korrekturen oder direkt fiktive Soll-Bahnpunkte (Anweisungen) eingebbar sein.

Es wird also eine Schnittstelle benötigt, über die ein Datenaustausch im Takt der Lageregelung erfolgen kann. Theoretisch ist es möglich, bei Bahnsteuerungs- oder Vorsteuerungsverfahren die durch Modifikation der Soll-Bahnpunkte erzeugten Bewegungsanweisungen bereits im Roboterprogramm abzulegen. Dies erfordert allerdings die Speicherung der Bahnen im Takt der Lageregelung und nicht, wie es üblich ist, nur die Stützpunkte von abschnittsweise linearen oder zirkularen Bahnen.

[3]Mit Referenztrajektorie ist hier nicht, wie sonst, die nominelle Soll-Trajektorie gemeint sondern eine Zwischentrajektorie. Der Begriff wird in der Literatur zur prädiktiven Regelung in dieser Bedeutung gebraucht, im Gegensatz zum Sprachgebrauch dieser Arbeit.

Weiterhin sollte eine industrielle Steuerung die aktuellen Positionen zu jedem Abtastschritt zur Verfügung stellen, z. B. in Form von gemessenen Gelenkwinkeln. Nicht ausreichend ist dagegen der rechnerisch ermittelte Zustand, der durch Filterung der Anweisungen innerhalb der Steuerung berechnet wird und irreführend als Ist-Position bezeichnet wird. Einige Sensorschnittstellen, z. B. [138] liefern nur diesen Zustand, was keine Aussagen über die Dynamik des Roboters erlaubt und somit nicht zur Regelung verwendet werden kann.

Auch bei Bahnsteuerung oder Vorsteuerung ohne Messwertrückführung (siehe Abschnitt 4.3.3) sollte der aktuelle Zustandsverktor außerhalb der industriellen Steuerung (unterste Ebene der hierarchischen Struktur) verfügbar sein, dann aber, abgesehen von der Bahnplanung, nur in der Adaptionsphase, also offline. Danach werden in der mittleren Ebene der hierarchischen Struktur keine Messwerte mehr verarbeitet. Ob die Bahnplanung den aktuellen Zustand des Roboters benötigt, hängt von der Art des Planungsverfahrens ab (siehe Abschnitt 4.4).

Das vorgeschlagene adaptive Verfahren arbeitet im Roboterkonfigurationsraum. Es werden also Gelenkvariablen vorgegeben und aktuelle Ist-Werte erwartet. Bei bekannter Kinematik des Roboters kann aber auch eine kartesische Schnittstelle sinnvoll sein, da sich alle Werte umrechnen lassen. Ebenso können die Anweisungen angepasst werden, wenn die Industrieroboterschnittstelle anstelle von Roboterpositionen Robotergeschwindigkeiten verlangt. Geschwindigkeitsanweisungen werden als Differenz von aufeinander folgenden Positionsanweisungen berechnet.

Steuerungsbedingte Totzeiten spielen abgesehen von der Messwertrückführung und der online Bahnplanung keine Rolle, da die aktuellen Ist-Werte dann nur offline zur Adaption verwendet werden. Bei offline Berechnungen kann die Totzeit vollständig berücksichtigt werden. Wenn die Ist-Werte allerdings online benötigt werden, sollte die Totzeit aus Stabilitätsgründen möglichst gering sein.

Schließlich vereinfacht es die Struktur des adaptiven Systems und erhöht die Genauigkeit, wenn die Kopplungen zwischen den Achsen gering sind. Dies ist bei Robotern der Fall, bei denen die Motorträgheitsmomente dominant gegenüber den Armträgheitsmomenten sind. Bezüglich der letzten Annahme kann festgestellt werden, dass sie das adaptive System vereinfacht, die Bahntreue aber nicht grundsätzlich begrenzt.

Hervorzuheben bleibt, dass keine Momenten- oder Stromschnittstelle zur Reali-

sierung des adaptiven Verfahrens benötigt wird. Es wird auch kein vorgegebenes Modell oder inverses Modell des Roboters vorausgesetzt. Sowohl von der Kenntnis des Systems als auch von den Hardwareschnittstellen benötigt das Verfahren also wesentlich weniger als rein modellgestützte Ansätze.

4.3.3 Struktur des adaptiven Systems

In Erweiterung der Aufgabenstellung aus Abschnitt 4.3.1 wird ein System ausgewählt, das verschiedene Betriebsarten vorsieht (siehe auch Tabelle 4.1):

1. Bahnsteuerung von speziell trainierten festen Trajektorien
2. Vorsteuerung von während der Adaptionsphase noch unbekannten Bahnen
3. Vorsteuerung von während der Adaptionsphase unbekannten Bahnen und Messwertrückführung zur Kompensation von stochastischen Störungen

Dabei entspricht Betriebsart 2 der im Abschnitt 4.3.1 geforderten Methode. Eine Erweiterung zu Betriebsart 3 ist leicht möglich und erlaubt unter Umständen eine weitere Erhöhung der Bahngenauigkeit. Betriebsart 1 beschreibt einen einfachen Sonderfall.

Es wird also zwischen der Steuerung von festen Trajektorien, die wiederholt abgefahren werden sollen, und einer allgemeinen Vorsteuerung von a priori unbekannten Bahnen unterschieden. Im ersten Fall erzeugt die Adaption eine Folge von Stellgrößen, die im Abtasttakt an den Roboter abgegeben werden. Dabei spricht man auch von Trajektorienlernen.

In den anderen Fällen wird ein Regler adaptiert, der aufgabenunabhängig den Roboterzustand und die Soll-Bahn gewichtet, um daraus optimale Stellgrößen zu generieren. Dieser Regler entspricht einem inversen dynamischen Modell des geregelten Roboters. Er besteht zumindest aus einer Vorsteuerung, also einer Gewichtung der zukünftigen Bahnpunkte. Es wird unterschieden, ob eine Rückführung des Roboterzustands vorhanden ist (Betriebsart 3), oder ob eine reine Vorsteuerung (Betriebsart 2) gewählt wurde, da sich in beiden Fällen die Adaption unterscheidet. Dabei wird der Zustand durch die Ist-Werte der Roboterposition und die abgegebenen Kommandos zu verschiedenen Zeitpunkten repräsentiert. Durch die Rückführung des Roboterzustands lässt sich eine

Betriebsart	Bahnsteuerung	Vorsteuerung ohne Messwertrückführung	Vorsteuerung mit Messwertrückführung
Zahl der Trajektorien zur Adaption	1	≥ 1	≥ 1
Trajektorie nach Abschluss der Adaption	wie bei Adaption	frei wählbar	frei wählbar
Dynamische Bahnfehler	kompensiert	kompensiert	kompensiert
Totzeit in der industriellen Steuerung (abgesehen von Sensorrückkopplungen)	kein Einfluss	kein Einfluss	kein Einfluss
Erreichbare Bahngenauigkeit, wenn stochastische Störungen gering sind	sehr hoch	hoch	hoch
Erreichbare Bahngenauigkeit, wenn stochastische Störungen dominant sind	gering	gering	mittel
Schritte der Adaption	2	3	3
Adaption von	Folge von Kommandos	Parametern der Vorsteuerung	Parametern des Reglers (mit Vorst.)
Initialisierung durch	Soll-Bahn	0, außer für aktuellen Sollwert	0, außer für aktuellen Sollwert
Eingänge im online-Betrieb	-	Sollwerte	Sollwerte, Ist-Werte, bereits abgegebene Kommandos
Eingänge für 3. Adaptionsschritt	-	Sollwerte	Sollwerte, Ist-Werte, bereits abgegebene Kommandos
Prädiktion der Soll-Bahn	erforderlich	erforderlich	erforderlich
Einsatz von Sensoren zur Spezifikation der Soll-Bahn	nicht möglich	erlaubt	erlaubt

Tabelle 4.1: Eigenschaften der verschiedenen Betriebsarten

Erhöhung der Bahngenauigkeit erreichen. Allerdings zeigen die Untersuchungen aus Kapitel 3 eine geringe Empfindlichkeit gegenüber stochastischen Einflüssen, so dass der Vorteil der Zustandsrückführung bei den in dieser Arbeit untersuchten Systemen eher gering ausfallen wird.

Eine reine Messwertrückführung ist dagegen nicht vorgesehen. Die hohe Reproduzierbarkeit der Bewegungen beweist, dass die Kaskadenregelung nach Abb. 4.7 bereits genügend optimiert ist, sodass eine weitere Verbesserung der Bahntreue nur durch zusätzliche Informationen möglich ist, die der Robotersteuerung direkt nicht zur Verfügung stehen.

Da nach Abschnitt 4.3.1 kein genaues Modell des Roboters vorgegeben werden soll, ist eine adaptive Struktur vorteilhaft. In allen Fällen wird während der Adaption die jeweils benutzte Soll-Bahn in Form von Abtastwerten der Positionen im Roboterkoordinatensystem als bekannt angenommen. Aufgabe der Adaption ist die (prädiktive) Modifikation der kommandierten Trajektorie in eine fiktive Bahn, deren Verarbeitung durch die industrielle Steuerung und das dynamische System gerade die Ausführung der Soll-Bahn ergibt. Lernverfahrnen wie *reinforcement learning* oder *Q-learning* (vergl. Abschnitt 2.4.3) sind dagegen aufgrund der Kenntnisse über das Robotersystem nicht nötig und werden daher aufgrund des hohen Trainingsaufwands auch nicht verwendet.

Unter den in Abschnitt 2.4 aufgeführten Adaptionsmethoden sind nur Verfahren mit getrenntem Streckenmodell und Regler geeignet, um ein bestimmtes Verhalten vorzugeben. Deshalb wird die Struktur nach Abb. 2.12 (Adaption mit Identifikationsmodell nach Isermann) ausgewählt. Die Struktur nach Abb. 2.13 hat zwar auch ein getrenntes Modell, dort ist die Regleradaption aber fest vorgegeben (Minimierung des Trajektorienfehlers durch Backpropagation [212]), was auch keine Spezifikation des Regelkreises erlaubt.

Im Gegensatz zur Struktur müssen die von Isermann in [108] aufgeführten Reglerentwurfsmethoden verworfen werden, da sie primär die Rückkopplung von Messwerten durch Schließung des Regelkreises betrachten, was hier nicht verändert werden soll. Außerdem werden dort nur parametrische, meist lineare Regler entworfen.

Daher wird die Regleradaption entgegen Abb. 2.12 in zwei Funktionsblöcke aufgeteilt, nämlich einen Funktionsblock zur Festlegung der optimalen Stellgrößen und einen Funktionsblock, um diese Stellgrößen aufgrund der Eingangswerte zu erzeugen. Dieser zweite Funktionsblock wird an die Struktur des Reglers angepasst.

Insgesamt ergeben sich also drei Schritte der Adaption und zwar die Bildung eines Robotermodells, die a posteriori Bestimmung optimaler Bewegungsanweisungen für eine Trainingsbewegung und schließlich die eigentliche Regleradaption.

Diese Struktur erlaubt nun gegenüber den anderen Architekturen einen selektiveren Entwurf, sodass z. B. hochfrequente Soll-Bahnen nach Möglichkeit exakt ausgeführt werden, während die Adaption unempfindlich gegenüber hochfrequenten Messstörungen ist. Außerdem kann der noch näher zu spezifizierende

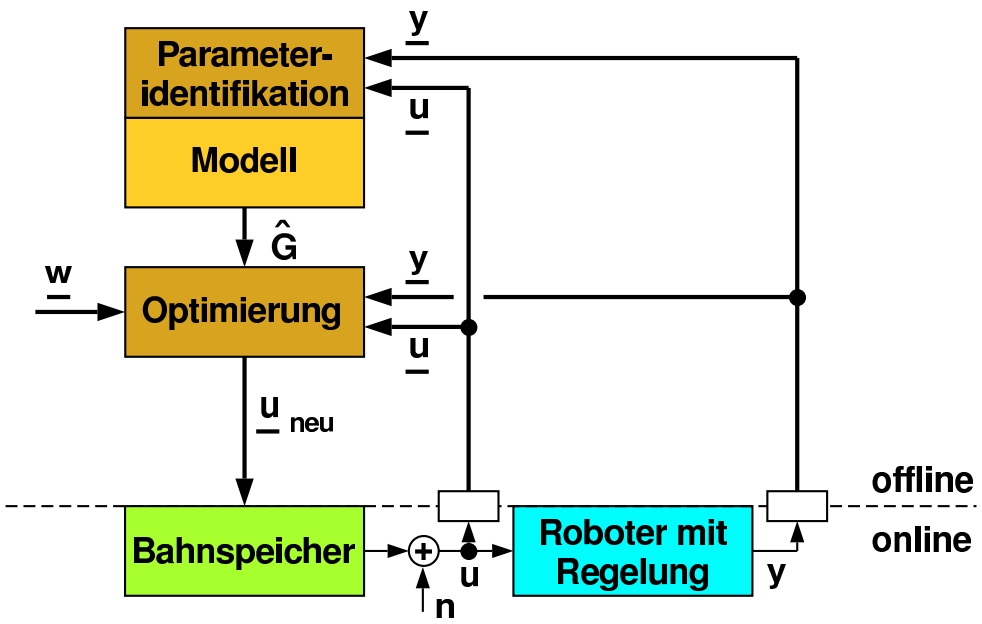


Abbildung 4.8: Struktur des adaptiven Verfahrens bei Bahnsteuerung (unterer Teil = online Steuerung, oberer (gelber/brauner) Teil = offline Adaption, leere Funktionsblöcke = Datenspeicher für gesamte Trajektorie, dicke Pfeile = Daten von mehreren Zeitpunkten)

Regler durch das identifizierte Modell inkrementell verbessert werden, was die Genauigkeit der Adaption erhöht (vergl. Abschnitt 2.4).

Die Strukturen der verschiedenen Betriebsarten sind in Abb. 4.8 bis 4.10 enthalten. Sie unterscheiden sich nur im unteren linken Bereich. Zur Darstellung wurden in der Regelungstechnik übliche Bezeichungen gewählt, da der Ansatz der Adaption nicht nur für die gestellte Aufgabe der Erhöhung der Bahngenauigkeit an positionsgeregelten Robotern gültig ist. Daher sind die Stellgrößen (Bewegungskommandos), die Messwerte (Encoderwerte) und die Sollwerte (Soll-Bahnpunkte) durch die Variablen u, y und w ausgedrückt. Die unterstrichenen Buchstaben $\underline{u}$, $\underline{y}$ und $\underline{w}$ bedeuten Variablen zu mehreren Zeitpunkten, i. A. von der gesamten Trajektorie. n beschreibt eine stochastische Anregung zur Überlagerung der Stellgröße, was allerdings nur zur Parameteridentifikation benötigt wird. Der Funktionsblock „Roboter mit Regelung“ enthält neben dem physikalischen Roboter auch noch die standardmäßige industrielle Steuerung (Kaskadenregelung).

Alle drei Strukturbilder enthalten im unteren Bereich den online Teil. Dies ist die Steuerung des Roboters bzw. der Regelkreis. Alle Daten der Regelung

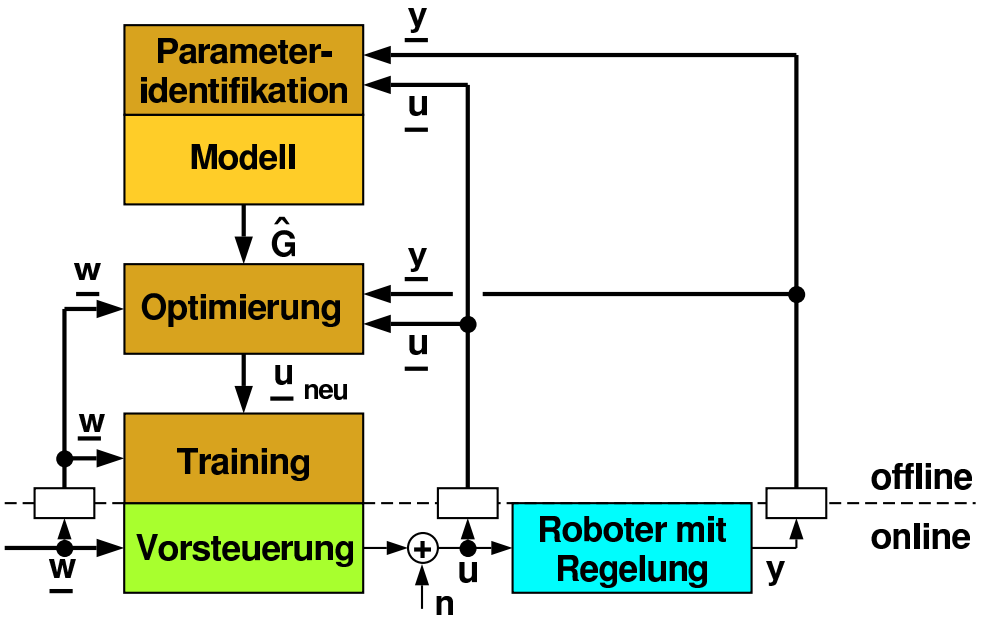


Abbildung 4.9: Struktur des adaptiven Verfahrens bei Vorsteuerung (unterer Teil = online Vorsteuerung, oberer (gelber/brauner) Teil = offline Adaption, leere Funktionsblöcke = Datenspeicher für gesamte Trajektorie, dicke Pfeile = Daten von mehreren Zeitpunkten)

/ Steuerung werden aufgezeichnet und nach Beendigung der Roboterbewegung offline verarbeitet, was in Abschnitt 4.2.3 durch gelbe Funktionsblöcke „Regleradaption“ zusammengefasst wurde. Hier wird die Adaption genauer ausgeführt. Dabei beschreiben die Funktionsblöcke Parameteridentifikation, Optimierung und Training die verschiedenen Schritte der Adaption. Die Wissensbasen sind Modell und Bahnspeicher, Vorsteuerung oder Regler. Die Darstellungen betrachten nur jeweils ein Gelenk. Die Adaption erfolgt also getrennt für jede Achse des Roboterkoordinatensystems. Dies ist zulässig, da die Kopplungen bei Betrachtung im Gelenkraum in erster Näherung vernachlässigbar sind.[4]

Die Adaption lässt sich in mehreren Schritten beschreiben. Grundlage ist die Identifikation eines Modells des dynamischen Verhaltens des positionsgeregelten Roboters. Dieses Modell wird in einem ersten Schritt bestimmt. Im zweiten Schritt werden unter Nutzung dieses Modells a posteriori die für eine abgefahrene Bahn optimalen Stellgrößen u_{neu} generiert, mit denen die Bahn ideal ausgeführt worden wäre. Das heißt, dass die aufgezeichneten Stellgrößen u aufgrund der Regeldifferenzen $w - y$ modifiziert werden. Bei Bahnsteuerung für eine feste Trajektorie (Betriebsart 1) wird diese ideale Stellfolge direkt im

[4]Später wird eine Entkopplung integriert, die eine gelenkübergreifende Struktur erfordert.

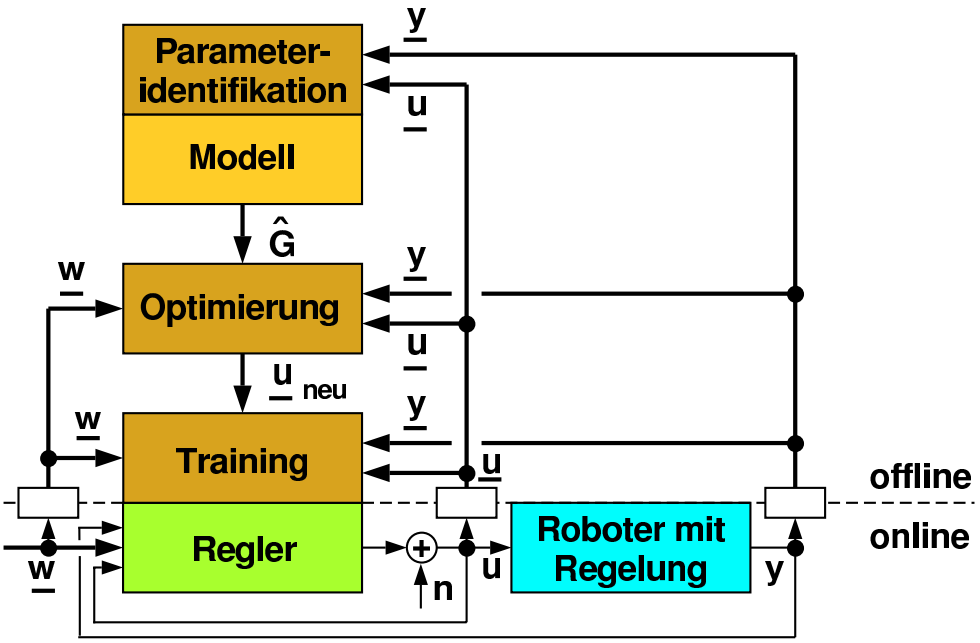


Abbildung 4.10: Struktur des adaptiven Verfahrens bei Regelung mit integrierter Vorsteuerung (unterer Teil = online Regelung (inkl. Vorsteuerung), oberer (gelber/brauner) Teil = offline Adaption, leere Funktionsblöcke = Datenspeicher für gesamte Trajektorie, dicke Pfeile = Daten von mehreren Zeitpunkten)

Bahnspeicher abgelegt. Bei Vorsteuerung bzw. Vorsteuerung mit Rückführung des Roboterzustands müssen dagegen noch die Vorsteuerung bzw. der Regler bestimmt werden. Diese werden so adaptiert, dass sie die soeben ermittelten optimalen Bewegungsanweisungen aus den Soll-Bahndaten und ggf. den Ist-Zuständen rekonstruieren. Bei ausreichendem Training können dann auch bei untrainierten Trajektorien geeignete Bewegungsanweisungen generiert werden.

Die gewählten Darstellungen erlauben, soweit es sich nicht um eine Bahnsteuerung handelt, eine prädiktive Regelung, da eine Vorsteuerung aufgrund zukünftiger Soll-Bahnpunkte vorgesehen ist. Nur dadurch ist entgegen reinen Messwertrückführungen oder kausalen Filterungen der Bahn eine Kompensation der dynamischen Fehler möglich. Aus diesem Grund ist keine Betriebsart mit Messwertrückführung ohne Vorsteuerung vorgesehen.

Aufgrund der Adaption wird ein Verhalten erreicht, das für die Trainingsdaten (nahezu) optimal ist. Es kann also kein robustes Verhalten für alle möglichen (untrainierten) Arbeitszustände erreicht werden. Der Unterschied besteht darin, dass ein optimaler Regler ein Gütemaß für die trainierten Trajektorien mini-

miert, während ein robuster Regler für alle möglichen Arbeitspunkte zumindest stabil, i. A. aber für keinen Arbeitspunkt gleichzeitig auch optimal ist.

Optimale Regler sind i. A. nicht robust, sondern empfindlich gegenüber Parameterschwankungen oder unmodellierten Nichtlinearitäten [86]. Durch Adaption kann man aber keinen robusten Regler bestimmen, da ohne Systemkenntnis unbekannt ist, ob ein Trainingsdatensatz auch alle möglichen Parametervariationen enthält. Ohne Systemkenntnis ist auch keine Empfindlichkeitsanalyse möglich. In der Praxis geht man davon aus, dass ein adaptierter Regler für einen Arbeitsbereich robust ist, wenn das Training in allen relevanten Zuständen erfolgt. Eine geringe Empfindlichkeit des adaptierten Reglers gegenüber Störungen oder ungünstigen Parametern des adaptiven Systems wird weiterhin erreicht, indem zur Adaption Schätzverfahren (Kalman Filter) verwendet werden und bei der Optimierung keine direkte Inversion des identifizierten Modells vorgenommen wird.

4.3.4 Erweiterte Architektur für elastische Roboter

Bei handelsüblichen Industrierobotern ist die Elastizität auf die Gelenke beschränkt, da die Armelemente starr ausgelegt sind. Die Elastizitäten der Robotergelenke bewirken einerseits niederfrequente Abweichungen zwischen den antriebs- und abtriebsseitig gemessenen Bahnen, andererseits können auch hochfrequente Schwingungen auftreten. Dadurch ergeben sich Einschränkungen des durch das adaptive System erreichbaren Verbesserungspotentials.

In dieser Arbeit wird angenommen, dass die Bahnplanung eine stetige Bahn generiert, die keine Eigenschwingung des Roboters anregt. Dies hat sich im Experiment bestätigt, sofern nicht versucht wird, sensorgeregelt mit zu hoher Geschwindigkeit zu fahren. Auch industrielle Steuerungen begrenzen die Bandbreite der Trajektorien, um Schwingungen zu vermeiden (z. B. [137]).

Behandelt werden also Bahnabweichungen, die sich mit einer Abtastzeit von etwa 10 ms kompensieren lassen. Mit dieser Einschränkung kann die in Abschnitt 4.3.3 vorgeschlagene Struktur auch auf elastische Systeme angewendet werden. Unterschiedlich ist lediglich die Quelle der Ist-Werte y bei der Adaption.

Es ist ausreichend, die Annahme von Elastizität in den Gelenken auf deren Freiheitsgrad zu beschränken. Dadurch erhält man eindimensionale elastische

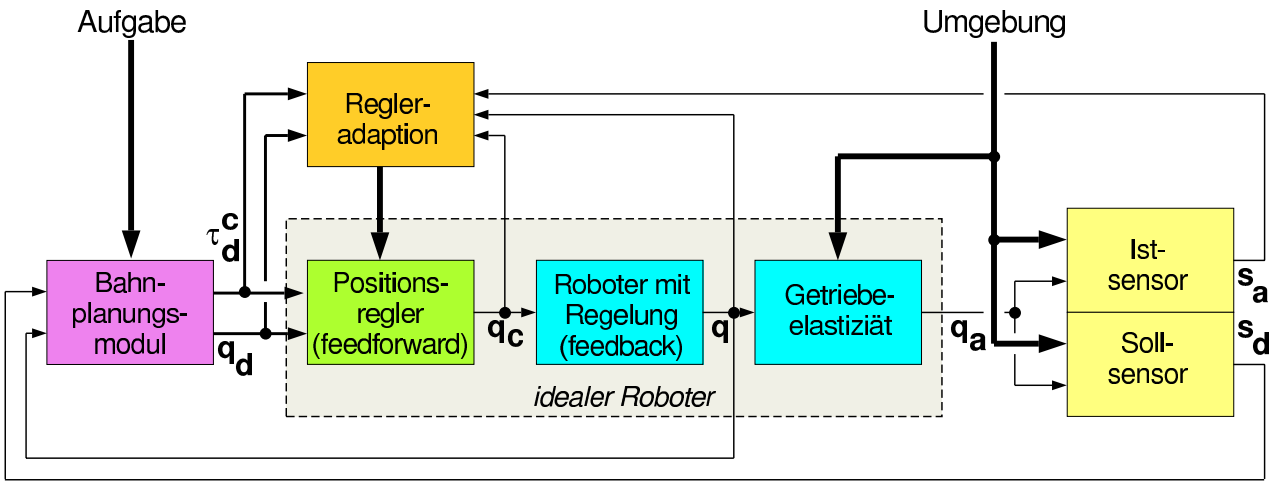


Abbildung 4.11: Regelkreisstruktur mit externer Messung der Ist-Position

Getriebemodelle nach Abb. 3.9. Entsprechend dieser Notation unterscheidet Abb. 4.11 zwischen den antriebsseitigen Gelenkwinkeln $\mathbf{q}$ und den durch abtriebsseitige Gelenkwinkel $\mathbf{q}_a$ beschriebenen Armelementpositionen. Dabei werden die antriebsseitigen Werte durch Encoder an den Motoren gemessen und dann mit dem Übersetzungsverhältnis der Getriebe skaliert. Der *ideale Roboter* hat die tatsächliche Position des Armelements als Ausgang und nicht die durch die Motorposition genäherte. Daher sollten die Armelementpositionen $\mathbf{q}_a$ zur Bahnplanung und zur Adaption zurückgeführt werden. Dies ist aber nicht möglich, da diese Werte direkt nicht messbar sind, sondern erst sensorisch erfasst werden müssen.

Den gleichen Sachverhalt weist die erweiterte hierarchische Architektur nach Abb. 4.12 auf. Auch hier ist gegenüber Abb. 4.6 ein zusätzlicher Eingang der Regleradaption vorgesehen, der die Anteile der Ist-Position bereitstellt, die wegen elastischer Effekte nicht direkt messbar sind. Die Positionen der Armelemente $\mathbf{q}_a$ sind in Abb. 4.12 aber nicht vorhanden. Das liegt daran, dass sie als Signal nur implizit in Form der Sensordaten vorliegen. Sie sind in Abb. 4.11 aufgeführt, um den Einfluss der tatsächlichen Armpositionen auf die Sensoren darzustellen.

In beiden Abbildungen 4.11 und 4.12 wird auf der Sensorseite unterschieden, ob die Sensorik die gewünschte Bahn ermittelt (Soll-Sensor), oder ob es sich um eine Vermessung der Ist-Bahn handelt (Ist-Sensor). Die Unterscheidung ist nötig, um das dynamische Verhalten des Roboters von speziellen Eigenschaften der Soll-Bahn, z. B. Krümmungen, zu trennen. Auf die Bewegung des Roboters kann Einfluss genommen werden, die Soll-Bahn gilt dagegen als feste Vor-

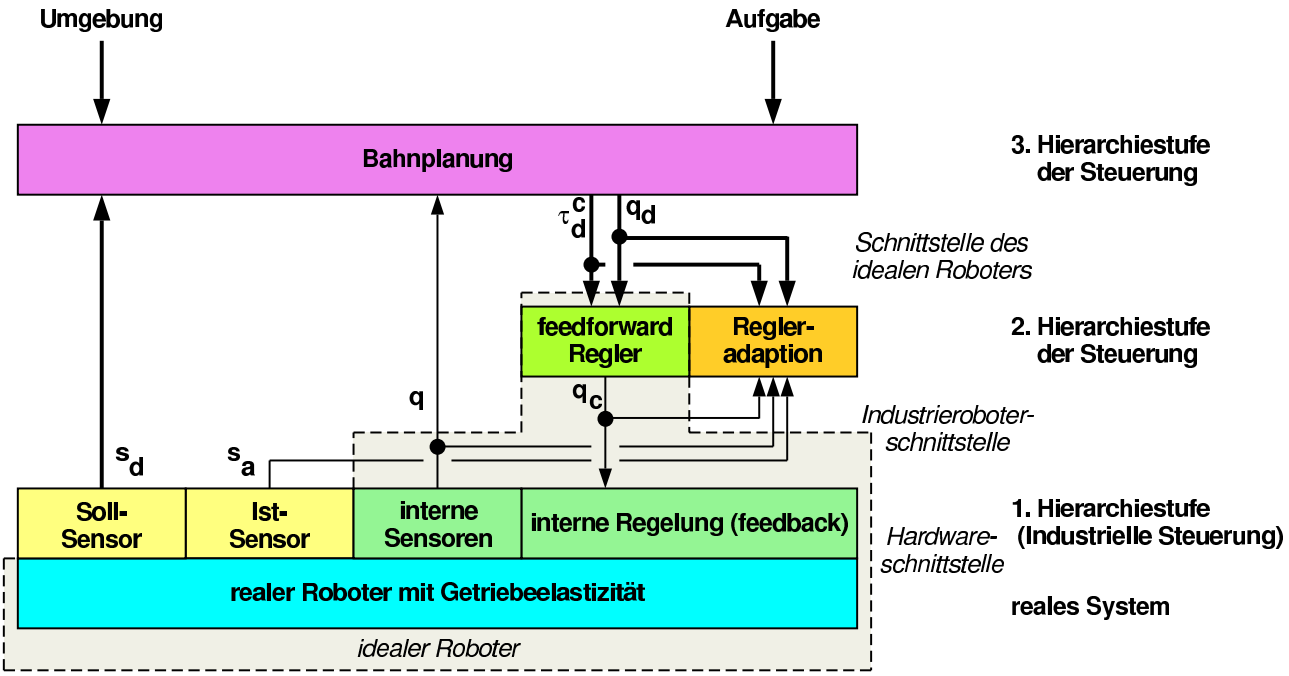


Abbildung 4.12: Hierarchisch gegliederte Struktur mit externer Messung der Ist-Position

gabe. Außerdem entspricht diese Aufspaltung den beiden oberen Ebenen des hierarchischen Systems. Zur Analyse ob die Sensorrückkopplung oder die Regelung verbessert werden muss, wird die Schnittstelle zwischen beiden Systemen benötigt. Wie gut die erreichte Bahngenauigkeit zu beurteilen ist, hängt bei gleichem numerischen Bahnfehler davon ab, wie leicht die Soll-Bahn auszuführen ist.

Diese aktuelle Position y des Roboters wird durch die an den Motoren gemessenen Gelenkwinkel $\mathbf{q}$ repräsentiert. Sie können durch Sensorwerte $\mathbf{s}_a$ ergänzt werden, wenn eine externe Erfassung der Position möglich ist [154]. $\mathbf{s}_a$ liegt i. A. als kartesischer Messvektor vor, meist beschränkt auf die 3 Positionskoordinaten, also ohne Orientierungsanteil. Wenn die inverse kinematische Transformation nicht eindeutig ist, können in dem Funktionsblock Regleradaption die Gelenkwinkel auch in kartesische Werte transformiert und durch die Sensorwerte ersetzt werden.

Die Adaptionskomponente modifiziert dann die Vorsteuerungen der einzelnen Gelenke auf der Basis kartesischer Korrekturinformation. Im Fall eines asynchronen Ist-Sensors ist es sinnvoll, die Transformation der Ist-Werte auf ein zeitvariantes Koordinatensystem durchzuführen sowie die Bewegungsrichtung anzugeben. Dabei sollten nur die antriebsseitigen Messwerte verwendet werden, da die abtriebsseitigen Werte bezüglich der Abtastzeit unsicher sind.

Der Aufwand der Sensorauswertung kann reduziert werden, indem die zeitliche Funktion der Differenzen zwischen antriebsseitig und abtriebsseitig gemessenen Positionen zur Kalibrierung verwendet wird. Bei weiteren Trainingsläufen werden die antriebsseitigen Positionen mit diesen Differenzen korrigiert und entsprechen dann den tatsächlichen Positionen.

Die bahnabhängige Korrektur ist nur dann zulässig, wenn die Bahn nicht wesentlich verändert wird.

4.4 Vorschlag für eine sensorgestützte Anpassung der Bahn

In diesem Abschnitt werden mehrere Ansätze zu einem Entwurf der obersten Ebene der hierarchischen Steuerungsstruktur beschrieben. Dabei werden die Möglichkeiten, die sich aus den Vorschlägen in den Abschnitten 4.2 und 4.3 ergeben, veranschaulicht werden.

4.4.1 Anforderungen

Voraussetzung für die vorgeschlagene Art der Sensordatenverarbeitung ist die Schnittstelle zu dem *idealen Roboter*, wie sie bereits beschrieben wurde. Die aktuelle Position muss hierzu zu den Abtastzeiten der Regelung verfügbar sein. Sollwerte, bestehend aus Position und ggf. Kontaktkraft, müssen ebenfalls vorgegeben sein.

Die Sensorik muss hierzu die gewünschte Bewegung online erfassen können. Dies ist z. B. dann der Fall, wenn durch ein Szenenbild eine fehlerhafte Werkstücklage erkannt wird und eine vorgegebene Bahn mit allen Abtastpunkten verschoben werden muss.

Solche Sensoren müssen kalibriert sein, um Positionen oder Positionsdifferenzen bestimmen zu können. Als sinnvoll erweist sich auch eine Prädiktion künftiger Sensorwerte, sei es durch Messung mit verteilten Sensoren, durch globale Messung oder durch geschickte Extrapolation der Werte.

Eine sensorgestützte Anpassung einer vorgegebenen Bahn soll unter diesen Voraussetzungen eine Soll-Bahn und ggf. die Soll-Kräfte vorgeben können, die eine

wie auch immer geartete Beschreibung einer Aufgabe erfüllen.

Diese Beschreibung kann symbolisch erfolgen oder durch einen funktionalen Zusammenhang wie eine programmierte Nominalbahn. Auch Relationen wie Abstände und Orientierungen vom Endeffektor zu Werkstücken oder Folgebewegungen (Tracking) sind geeignet. Dabei werden die zu adaptierenden Größen meist in Sensor- oder Werkstückkoordinaten angegeben.

Regelung und Bahnplanung sollten dabei getrennt erfolgen. Ist dies nicht möglich, könnte man eine direkte Sensorrückführung vorsehen, ohne Verwendung der Position als Zwischengröße (wie z. B. [152]). Mit solch einer Konfiguration muss allerdings das gesamte System neu adaptiert werden, wenn ein neuer Sensor eingesetzt werden soll oder wenn ein Roboter durch ein anderes Gerät gleichen Typs ersetzt wird. Sofern die adaptive Komponente implizit die Form der vorgegebenen Bahnen erfasst, ist sogar bei jeder neuen Aufgabe eine Anpassung nötig. Dies betrifft z. B. kraftgeregelte Konturverfolgungen entlang konvexer Kanten, da bei direkter Sensorrückführung die Krümmung solcher Bahnen mit den zukünftigen Bahnfehlern korreliert. Die Vorteile einer direkten Sensorrückführung sind, dass die Kalibrierung der Sensorik entfallen kann und bei unvollständiger Schnittstelle zu dem *idealen* Roboter (weil keine Prädiktion der Soll-Bahn oder der Soll-Kräfte möglich ist) ein besseres Verhalten erreicht werden kann.

4.4.2 Verarbeitung von Sensordaten

Die allgemeine Struktur (Abb. 4.13) für die Bahnplanung aus Abb. 4.6 und Abb. 4.12 wird im Folgenden entworfen und dann am Beispiel spezieller Aufgaben diskutiert. Als Beispiele werden Bahnplanung ohne Sensoren (Abb. 4.14), Kraftregelungen (Abb. 4.15) und der Einsatz von Bildverarbeitungssystemen zur Bahnplanung (Abb. 4.16) betrachtet.

In allen Fällen wird eine, bei industriellen Aufgaben immer vorgegebene, programmierte Bahn $\mathbf{x}_p$ angenommen, aus der sich die Soll-Bahn $\mathbf{x}_d$ für den *idealen Roboter* ergibt. Parallel werden die Sensordaten verarbeitet, die schließlich die Kompensation des dynamischen Einflusses der (Soll-)kontaktkräfte erlauben.

Aufgrund der Schnittstelle zu dem *idealen Roboter* können sowohl die Bahndaten als auch die Sensorwerte in jedem Abtastschritt auch für zukünftige Ab-

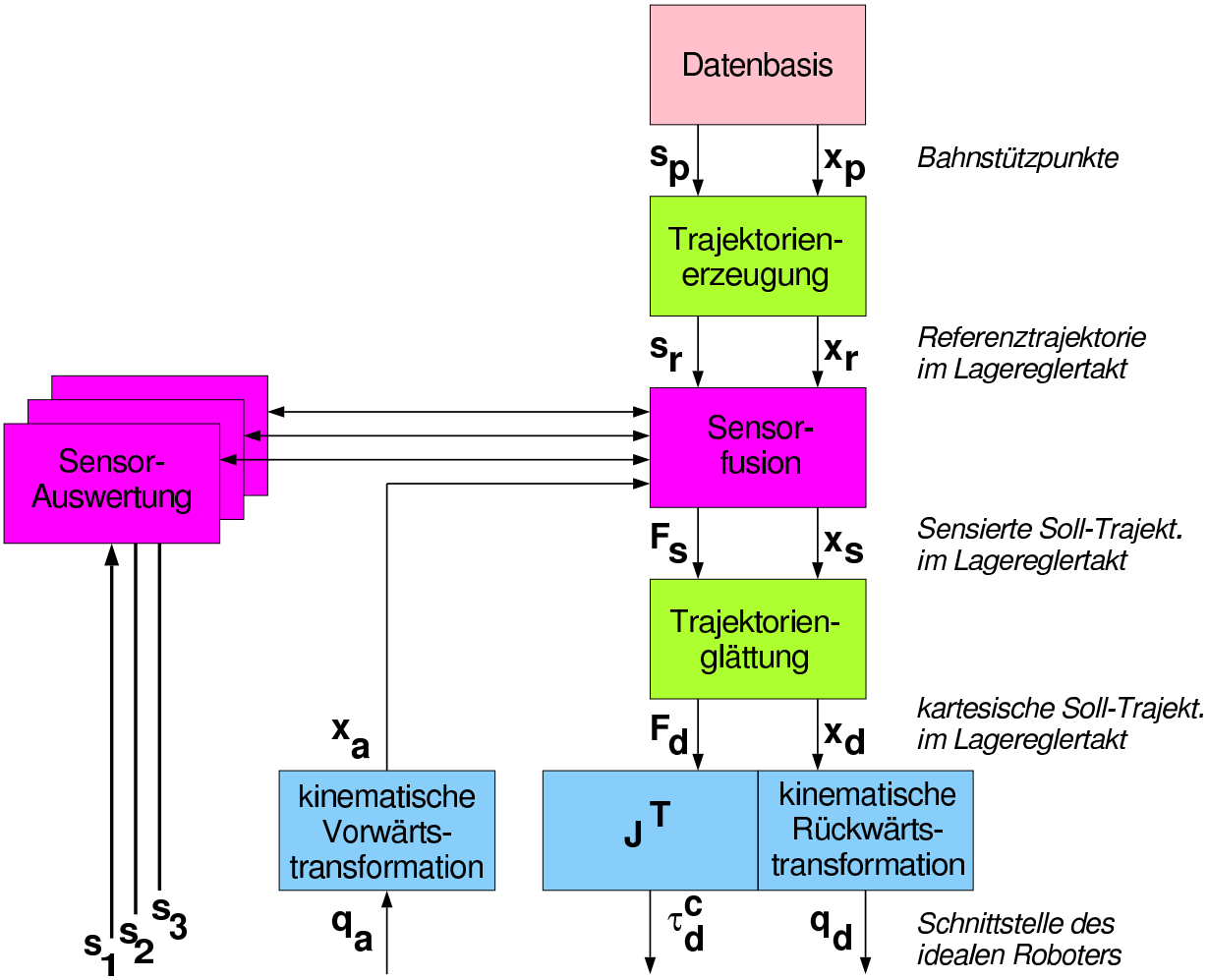


Abbildung 4.13: Struktur des Bahnplanungsmoduls der multisensoriellen Architektur aus Abb. 4.12

tastschritte bereitgestellet werden.

Hierzu sind einige Transformationen nötig, die in den Abbildungen blau unterlegt sind. Sie transformieren zwischen Achswerten $\mathbf{q}$ und kartesischen Positionen $\mathbf{x}$, sowie zwischen Kräften in Roboterbezugskoordinaten (Gelenkmomenten) τ^c und kartesischen Kräften und Momenten $\mathbf{F}$. Dabei besagt der Index c bei τ^c, dass es sich um den Einfluss der Kontaktkräfte handelt, also nicht um das gesamte durch den Antrieb aufzubringende Gelenkmoment.

In den grünen Funktionsblöcken werden Interpolationsaufgaben durchgeführt. Im einen Fall erzeugen sie aus der ggf. nur durch Stützpunkte beschriebenen programmierten Bahn $\mathbf{x}_p$ eine kontinuierliche Referenzbahn, die durch die Positionen $\mathbf{x}_r$ und die Sensorwerte $\mathbf{s}_r$ zu den Abtastzeitpunkten definiert ist. Im anderen Fall glätten sie die durch Sensordaten ggf. stark modifizierte Bahn,

sodass sie von einem Roboter ausführbar wird. Dabei werden insbesondere Beschleunigungsspitzen durch glatte Bewegungen ersetzt.

An dieser Stelle muss bemerkt werden, dass der Begriff „Referenztrajektorie" in dieser Struktur anders verwendet wird als bei der prädiktiven Regelung (vergl. [110]). Dort führt die Referenztrajektorie von einer aktuellen Ist-Position zu einem zukünftigen Punkt der Soll-Bahn. In der betrachteten Steuerungsstruktur stellt die Referenzbahn dagegen die nominelle Soll-Bahn dar, die durch Sensordaten verfeinert wird. Es soll also die Soll-Trajektorie ausgeführt werden und nicht die Referenztrajektorie.

Die eigentliche Modifikation der Bahn aufgrund von Sensordaten erfolgt in den violetten Funktionsblöcken. Sie ist stark aufgabenspezifisch. Allgemein gilt, dass jeder Sensor eine Vorverarbeitung benötigt, die eine oder mehrere Positionsdifferenzen generiert. Die Informationen der einzelnen Sensoren werden im Funktionsblock Sensorfusion zusammengefasst in die Robotersteuerung integriert. Dabei werden die Sensordaten in einigen Komponenten ausgewertet oder mit der jeweiligen Genauigkeit gewertet. Dadurch wird die Referenzbahn modifiziert. Das Ergebnis wird als sensorische Soll-Bahn $\mathbf{x}_s$ bezeichnet.

Bei mehreren Sensoren oder, im Fall der Bildauswertung, bei mehreren erfassten Merkmalen oder extrahierten Eigenschaften der Szene ist ein Schätzverfahren zur Bestimmung der sensorischen Soll-Bahn naheliegend, da dabei sich widersprechende Informationen mit ihrer Zuverlässigkeit bzw. Genauigkeit gewichtet werden können [155]. Aber auch andere Ansätze wie Voting oder Fuzzy Systeme zur Fusion sind möglich [135].

In den violetten Funktionsblöcke werden die aktuellen Positionen $\mathbf{x}_a$ verarbeitet. Bei Robotern ohne Getriebeelastizität entsprechen sie den Positionen $\mathbf{x}$, die aus den antriebsseitig gemessenen Achswerten $\mathbf{q}$ berechnet werden.

Es wurde bereits erwähnt, dass hochdynamische Bahnen in den grünen Funktionsblöcken geglättet werden, wenn die sensorische Soll-Trajektorie unerwünschte Spitzen enthält. Eine weitere Glättung kann notwendig sein, wenn die zu verschiedenen Zeitpunkten geplanten Bahnen sich stark unterscheiden. Dies kann durch Sensorrauschen oder große Abstände zwischen zwei Sensorauswertungen kommen. Maßstab ist dabei nicht die hohe Genauigkeit der Sensorwerte sondern die dadurch bedingte Bahnkorrektur. Die Filterung zwischen zwei Berechnungen der Bahn erfolgt am besten schon bei der Sensorfusion, da die Genauigkeit der einzelnen Sensorsignale in später abgearbeiteten Funktionsblöcken nicht

mehr verfügbar ist.

Von der Aufgabendefinition im Programmspeicher (Datenbasis) bis zur Sensorfusion werden die gewünschten Sollwerte der Sensorsignale in gleicher Weise wie die der Positionen interpoliert. Bei der Sensorfusion und bei der Berechnung der Positionsdifferenzen in den Sensorblöcken, werden diese Sollwerte (Referenzsensordaten $\mathbf{s}_r$) benötigt. Ab der Sensorfusion werden in Abb. 4.13 keine Sensorsignale oder -sollwerte mehr verarbeitet. Lediglich die Kontaktkräfte werden zur Kompensation der durch sie verursachten dynamischen Fehler an den *idealen Roboter* weiter gereicht.

Abb. 4.14 zeigt, dass diese Architektur, die eigentlich zur Integration von Sensordaten entworfen wurde, sich auch zur Beschreibung der herkömmlichen Robotersteuerungen oder der in Kapitel 3 beschriebenen Anordnungen eignet.

Bei der Kraftregelung nach Abb. 4.15 gibt es gegenüber der allgemeinen Architektur (Abb. 4.13) zwei Unterschiede. Zum einen ist ein zusätzlicher Funktionsblock angegeben, der die Sensordaten aus dem Sensorsystem in das kartesi-

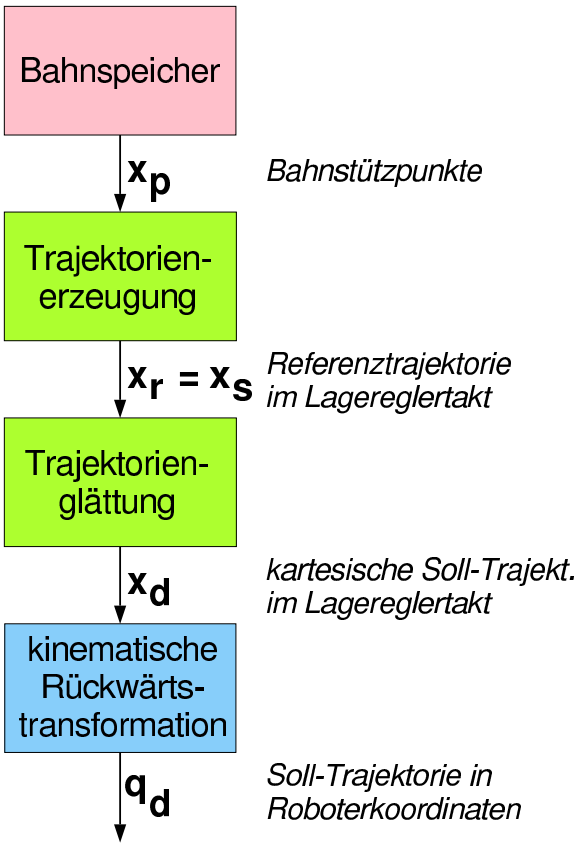


Abbildung 4.14: Steuerungsstruktur zur Bahnplanung ohne Sensor

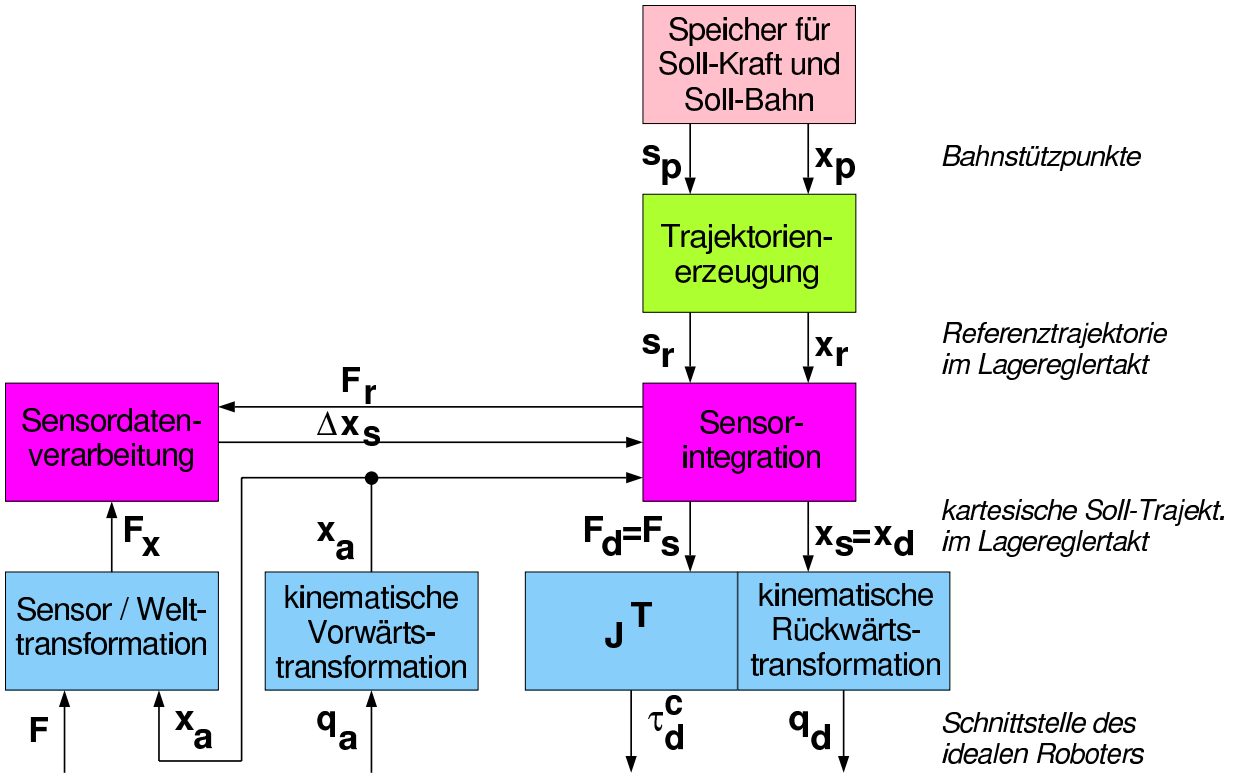


Abbildung 4.15: Steuerungsstruktur zur Kraftregelung

sche Inertialsystem transformiert. Zum anderen entfällt der Funktionsblock der Trajektorienglättung. Das kommt daher, dass ein Kraft- / Momentensensor wegen des beschränkten Wertebereichs i. A. so kleine Regeldifferenzen misst, dass der Roboter mit der Ausführung der daraus resultierenden Positionsänderungen keine Probleme hat. Umso wichtiger ist also, dass die Referenztrajektorie ausführbar ist.

Bei der Implementierung der violetten Funktionsböcke in Abb. 4.15 muss man unterscheiden, ob die Sensorwerte synchron zu jedem Abtastschritt vorliegen, oder ob der Sensor asynchron Werte liefert [210].

Im ersten Fall bildet der Sensorblock die Soll - Ist - Differenz und skaliert das Ergebnis mit der im System vorhandenen Elastizität.[5]

[5]Üblicherweise weisen Kraft- / Momentensensoren eine ausreichende Nachgiebigkeit auf, sofern das Messprinzip darauf beruht, kartesische Positionsänderungen zu erfassen, die bei der Ausübung von kartesischen Kräften auf eine definierte Nachgiebigkeit entstehen. Ohne solch eine Nachgiebigkeit im Sensor kann eine Elastizität des Roboters genutzt werden. Dann wird Gleichung (4.1) im Gelenkraum formuliert: $\Delta\mathbf{q}_s = \mathbf{J}^T \frac{\mathbf{F}_r - \mathbf{F}_s}{\mathbf{E}_q}$. Dabei ist $\mathbf{J}^T$ die transponierte Jacobi-Matrix und $\mathbf{E}_q$ enthält die Gelenkelastizitäten des Roboters. Ganz ohne Nachgiebigkeit ist keine Umrechnung von Kräften in Positionen möglich. Dann kann die vorgestellte Architektur nicht angewendet werden.

$$\Delta \mathbf{x}_s = \frac{\mathbf{F}_r - \mathbf{F}_x}{\mathbf{E}} \tag{4.1}$$

Die Sensorintegration addiert die gemessene Regeldifferenz zur aktuellen Position, ändert aber die Soll-Kräfte nicht. Dabei werden auch die Positionen nur in den Komponenten verändert, in denen der Sensor eine Beitrag leistet.

$$\mathbf{x}_s = \begin{cases} \mathbf{x}_r \\ \mathbf{x}_a + \Delta \mathbf{x}_s \end{cases} \tag{4.2}$$

$$\mathbf{F}_s = \mathbf{F}_r \tag{4.3}$$

Die Gleichungen (4.1) und (4.2) entsprechen der direkten Sensorrückführung in Gleichung (2.2). Dort wurde die gemessene Regeldifferenz $\Delta \mathbf{x}_s$ zur Berechnung eines neuen Positionskommandos gewichtet mit einer Reglerverstärkung zur zuletzt kommandierten Position addiert. Der gravierende Unterschied zu Gleichung (4.2) ist, dass die gemessene Regeldifferenz im multisensoriellen Ansatz zur *aktuellen* Position $\mathbf{x}_a$ addiert wird und nicht zur zuletzt *kommandierten* $\mathbf{x}_c$. Durch Addition der Relativmessung $\mathbf{F}_x$ zwischen Roboter und Umwelt zu der aktuellen Roboterposition ist $\mathbf{x}_s$ die sensorisch erfasste Soll-Position bzgl. der tatsächlichen Umwelt. Eine Reglerverstärkung oder die Verwendung zusätzlicher vorangegangener Differenzen ist hier nicht nötig.[6] Im Gegensatz dazu wird die aktuelle Roboterposition bei der direkten Sensorrückführung nicht verwendet. $\mathbf{x}_c$ in Gleichung (2.2) ist ein Kommando, das die gemessene Regeldifferenz unter Berücksichtigung der Roboterdynamik ausgleichen soll. Dabei wird $\mathbf{R} \cdot \Delta \mathbf{x}_s$, u. a. im Fall von Totzeiten in der Steuerung, mehrfach in fast gleicher Größe addiert, bevor der Roboter durch Bewegungsänderungen reagiert.

Eine dritte Art der Kraftregelung wird vorgeschlagen, wenn die Sensordaten asynchron erfasst werden, wenn also $\Delta \mathbf{x}_s$ nicht in jedem Regelungsschritt aktuell berechnet werden kann. Dabei wird angenommen, dass die Differenz $\Delta_r \mathbf{x}_s$ zwischen Referenzbahn und sensorischer Soll-Bahn sich nur langsam ändert. Zur Sensorintegration ist in diesem Fall eine Addition der Sensorkorrektur $\Delta_r \mathbf{x}_s$ sinnvoll.

Die Sensorintegration wird somit um

[6] Ein Regler anstelle von Gleichung (4.1) kann zur Prädiktion von sensorisch erfassten Positionen hilfreich sein, wenn der Sensor selbst nicht prädiktiv messen kann.

$$\mathbf{x}_s = \mathbf{x}_r + \Delta_r \mathbf{x}_s \tag{4.4}$$

und

$$\Delta_r \mathbf{x}_a = \mathbf{x}_a - \mathbf{x}_r \tag{4.5}$$

erweitert. In dem Sensorfunktionsblock wird $\Delta_r \mathbf{x}_s$ berechnet, wobei zur Sensorintegration ersatzweise Werte aus früheren Abtastzyklen genutzt werden, wenn die Sensorik nicht schnell genug abtasten kann. Zur Vermeidung von Inkonsistenzen wird die Sensorkorrektur

$$\Delta_r \mathbf{x}_s = \begin{cases} \mathbf{0} \\ \frac{\mathbf{F}_r - \mathbf{F}_x}{\mathbf{E}} + \Delta_r \mathbf{x}_a \end{cases} \tag{4.6}$$

auch bei Messung der einzelnen Koordinaten durch unterschiedliche Sensoren immer in allen Koordinaten bestimmt. Voraussetzung für die Sensorkorrektur ist, dass die Sensorverarbeitung die zum Zeitpunkt der Messung aktuelle Lage der Referenzbahn bezüglich der aktuellen Position kennt.

Der Ansatz der Gleichungen (4.1) und (4.2) ist formelmäßig identisch mit den Gleichungen (4.4) bis (4.6). Unterschiedlich ist einerseits, dass bei asynchron messendem Sensor die Sensorkorrektur $\Delta_r \mathbf{x}_s$ (sensorische Soll-Bahn - Referenzbahn) und nicht die gemessene Regeldifferenz $\Delta \mathbf{x}_s$ (Referenzsensorwert - Sensorwert) beibehalten werden. Dies ist sinnvoll, da die Regeldifferenz sich meist schneller ändert als die Unterschiede zwischen der offline geplanten Referenzbahn und der aufgrund der tatsächlichen Zustände der Umwelt modifizierten Soll-Bahn.

Andererseits wird in Gleichung (4.6) berücksichtigt, wenn die Positionen bei der Messung und bei der Sensorintegration sich aufgrund von Totzeiten bei der Signalverarbeitung unterscheiden. Die Sensordatenverarbeitung verwendet in Gleichung (4.6) immer den letzten Wert von $\Delta_r \mathbf{x}_a$, also, abgesehen von Totzeiten bei der Übertragung, den Wert bei der Messung.

Die Implementierung der Sensorintegration kann auch den Einsatz von Bildverarbeitungssystemen nach Abb. 4.16 unterstützen. Dies ist sinnvoll, da Bildverarbeitungssysteme typischerweise niedrige Abtastraten aufweisen.

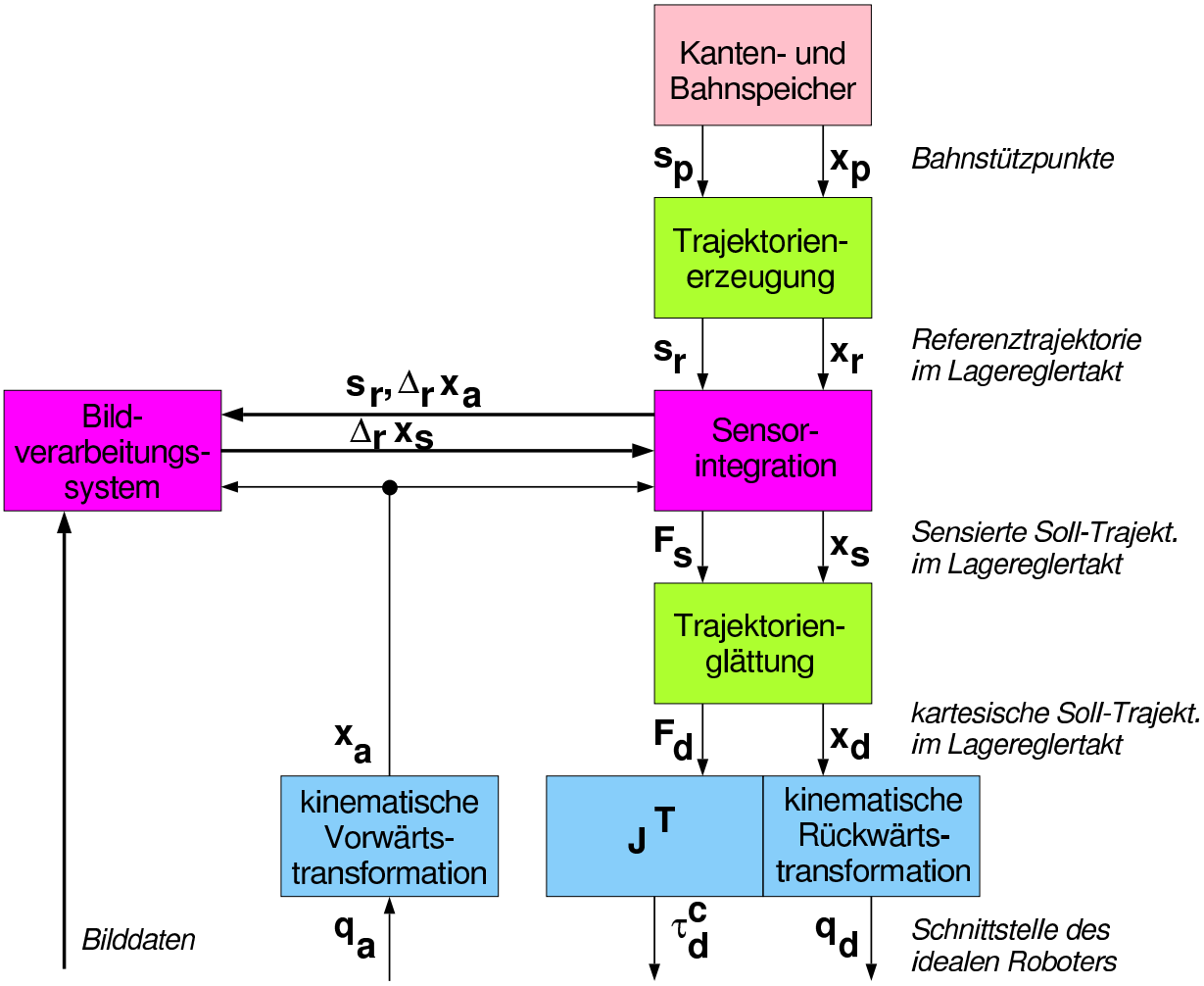


Abbildung 4.16: Erweiterung der Steuerungsstruktur zur Bahnplanung auf der Basis von Bilddaten

Darüber hinaus sollten zu jedem Zeitpunkt die Positionen und Achskräfte für mehrere zukünftige Zeitpunkte bereitgestellt werden können. Dies ist dann möglich, wenn das Sensorsystem prädiktiv arbeitet, also auch für zukünftige Punkte der Referenzbahn die Sensorkorrektur bestimmen kann.

Ein prädiktiver Kraftsensor ist z. B. ein Sensor, der in Bewegungsrichtung zu dem Werkzeug etwas entfernt ist und daher eine Kraft vor dem eigentlichen Kontakt mit dem Werkzeug vorhersagen kann. Derartige Sensoren können genutzt werden, um die Zustände der Umwelt für den Zeitraum der nächsten Abtastschritte zu beschreiben. Die zukünftigen erfassten und berechneten Soll-Positionen und Kraftsollwerte müssen dann durch Vergleich der Referenztrajektorie mit der sensorisch erfassten Umwelt berechnet werden. Die aktuellen Sensorsignale ergeben sich durch Vergleich der aktuellen Position mit der erfassten und berechneten Beschreibung der Umwelt.

Eine andere Art der Umwelterfassung ist bei Bildverarbeitungssystemen möglich. Dabei können ganze Szenen ausgewertet werden. Mit derart gebildeten Umweltmodellen können dann prädiktiv Soll-Bahnen bestimmt werden (siehe Anhang D.2).

Wenn es keine Möglichkeit der prädiktiven Umwelterfassung gibt, kann man auf eine Extrapolation bereits gemessener Bahnen zurückgreifen. Anhang D.1 beschreibt dies für eine Kraftregelungsaufgabe.

4.4.3 Erweiterte Steuerungsarchitektur für elastische Roboter

Elastizitäten wirken sich bei sensorgestützter Bahnplanung einerseits bei der Erfassung der Zustände der Umwelt und andererseits bei der Vorgabe der Soll-Positionen aus. Da die Schnittstelle des *idealen Roboters* die gewünschte Bahn des Endeffektors beschreibt und nicht die Bewegung der einzelnen Antriebseinheiten des Roboters,[7] ist die Steuerungsstruktur insbesondere an die geänderte Erfassung der Umwelt anzupassen. Dies bedeutet, dass die tatsächliche Ist-Position des Endeffektors ermittelt werden muss, da sie bei elastischen Systemen nicht direkt aus den Encoderwerten berechnet werden kann.

Im Folgenden wird zunächst der Unterschied zwischen Sensoren für Soll-Bewegungen und Sensoren für Ist-Bewegungen diskutiert. Sensorgestützte Anpassungen der Bahnen stützen sich ausschließlich auf Soll-Sensoren. Sie dienen nur zur Bahnplanung, wirken sich also nicht auf Funktionen in der mittleren Ebene der vorgeschlagenen hierarchischen Struktur aus. Insbesondere werden sie nicht bei der Adaption der Vorsteuerung benötigt.

Dagegen spielen die Sensoren für die aktuellen Positionen bei der Bahnplanung keine Rolle, wohl aber bei Adaptionsprozessen der mittleren Ebene. Andererseits benötigen die Sensordatenverarbeitungsprozesse die aktuellen Armpositionen nach Gleichung (4.2) oder (4.5).

Sofern geeignete Ist-Sensoren nach Abb. 4.11 zur Verfügung stehen, wird man sie wohl nutzen. In den meisten Fällen wird die tatsächliche Armposition aber nicht messbar sein. Das kann einerseits an Sensorschnittstellen liegen, die keine online Verbindungen zur Robotersteuerung vorsehen. Andererseits kann es

[7] Die gewünschte Bahn des Endeffektors wird zwar in Achskoordinaten ausgedrückt, es wird aber vorausgesetzt, dass dies die tatsächlichen Winkel zwischen den Roboterarmelementen sind und nicht die an den Encodern gemessenen antriebsseitigen Gelenkwinkel.

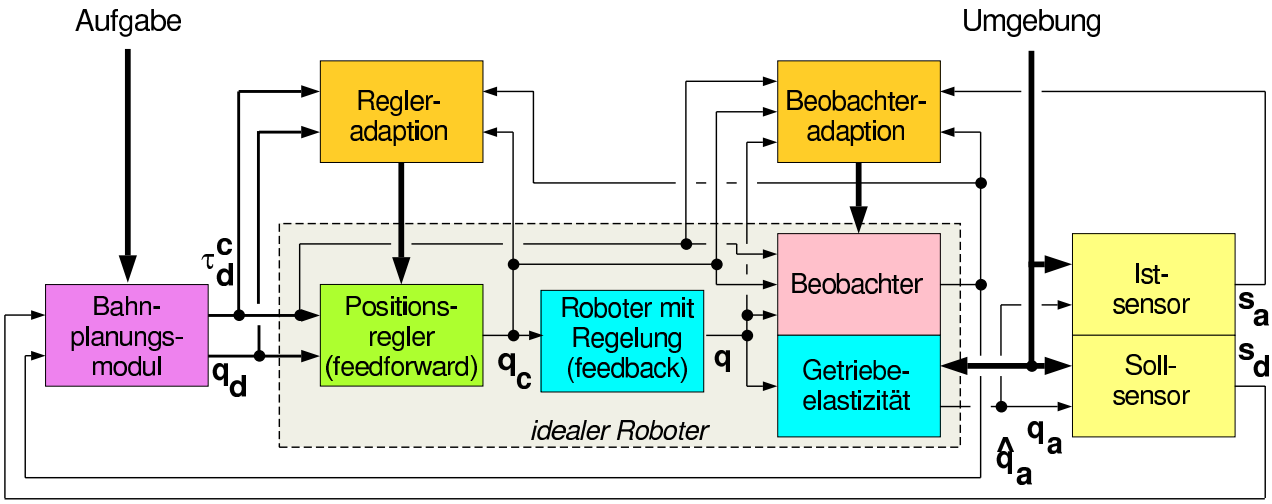


Abbildung 4.17: Regelkreisstruktur mit Beobachter für die Ist-Position

am Messprinzip der Sensoren liegen, das nur einen begrenzten Arbeitsbereich erlaubt.

Deshalb wird ein Zustandsbeobachter vorgeschlagen, der online die tatsächliche Armposition in allen Freiheitsgraden berechnet. (vergl. Abb. 4.17). Die Ausgangsdaten des Beobachters werden an all den Stellen verwendet, an denen die Position der Armelemente benötigt wird. Neben der Bahnplanung kann dies auch die Regleradaption sein, sofern der Ist-Sensor während der Adaption nicht verwendbar ist. Dagegen wirkt als Eingang der Sensoren nicht der beobachtete Wert sondern der tatsächliche Winkel des Armelements.

Bei der hierarchisch gegliederten Architektur nach Abb. 4.6 ergibt die Einführung eines Beobachters Abb. 4.18. Dabei wurde die Bahnplanung der Übersichtlichkeit halber in die Sensorauswertung (Perzeption) und die eigentliche Bahnplanung (Aktion) geteilt. Die Sensorauswertung stellt dabei u. a. die aus dem Ist-Sensor bestimmte kartesische Armposition $\mathbf{x}_a$ zur Verfügung.

Der Beobachter wird mit Hilfe des Ist-Sensors adaptiert. Hierzu wird eine geeignet angeregte Trajektorie gewählt. Zur Adaption des Beobachters muss der Ist-Sensor nicht zwingend online verfügbar sein.

Damit der Beobachter die Armbewegung mit ausreichender Genauigkeit voraussagen kann, muss er mit allen relevanten Informationen versorgt werden. Dies sind die auftretenden Beschleunigungen und der Zustand des Systems mit allen Parametern wie Armträgheitsmomenten oder Gelenkreibungskoeffizienten. Bei einem adaptiven System ergibt sich die gewünschte Abbildung, wenn die

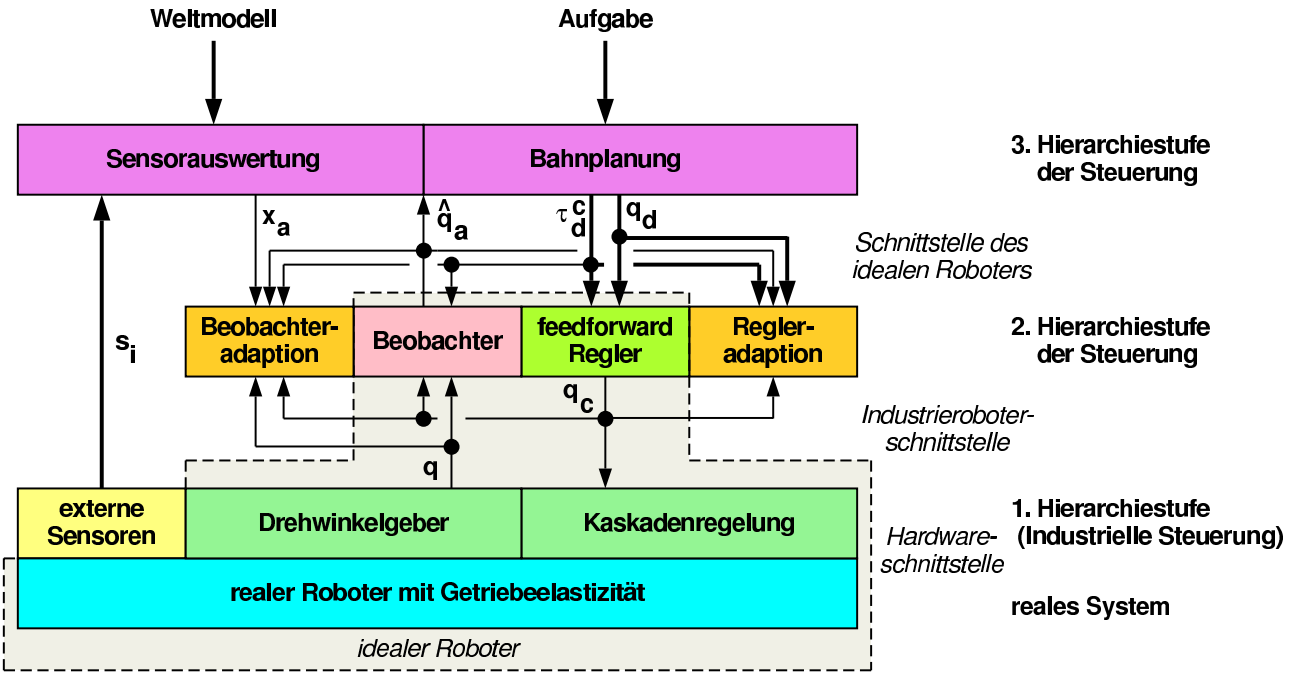


Abbildung 4.18: Struktur mit Beobachter für die Ist-Position

Auswahl der Eingangsgrößen ausreichend ist und die Struktur der Abbildung geeignet vorgegeben wurde.

Anstelle eines modellbasierten Ansatzes wird hier eine andere Realisierung des Beobachters vorgeschlagen. Näherungsweise kann man die Unterschiede zwischen antriebs- und abtriebsseitig gemessener Position duch ein Modell des Getriebes als Feder und Dämpfer beschreiben (siehe Abb. 3.9).

$$\tau = (\mathbf{q}_a - \mathbf{q}) \cdot \mathbf{E} + (\dot{\mathbf{q}}_a - \dot{\mathbf{q}}) \cdot \mathbf{D} \tag{4.7}$$

Dabei ist τ das Gelenkmoment, das die Positionsunterschiede bewirkt. τ kann von beiden Seiten des Getriebes berechnet werden. Von der Seite des Armes ergibt es sich aus der Beschleunigung und dem Armträgheitsmoment. Dabei erschwert es die Berechnung, dass die Beschleunigung des Armelements j in einem Inertialsystem auch von den Beschleunigungen der Gelenkwinkel der vorangehenden Gelenke abhängt und dass das wirksame Armträgheitsmoment von den Stellungen der weiteren Gelenke beeinflusst wird. Außerdem wirken Nutzlast und Kontaktkräfte auf den Arm.

Aus diesem Grund wird τ zunächst von der Motorseite berechnet.

$$\tau = \tau_{motor} - \mathbf{I}_r \cdot \ddot{\mathbf{q}} - Reibung \tag{4.8}$$

Dabei ist $\mathbf{I}_r$ das wirksame Trägheitsmoment der rotierenden Teile des Motors. τ_{motor} wird durch die Gelenkregler aufgrund der Kommandos $\mathbf{q}_c$ und von messbaren Größen eingestellt.

$$\tau_{motor} = \mathbf{f}(\mathbf{q}_c, \mathbf{q}, \dot{\mathbf{q}} \cdots) \tag{4.9}$$

Wenn man die Gleichungen (4.7) und (4.8) gleich setzt und τ_{motor} durch Gleichung (4.9) ersetzt, ergibt sich

$$\mathbf{f}(\mathbf{q}_c, \mathbf{q}, \dot{\mathbf{q}} \cdots) - \mathbf{I}_r \cdot \ddot{\mathbf{q}} - Reibung = (\mathbf{q}_a - \mathbf{q})) \cdot \mathbf{E} + (\dot{\mathbf{q}}_a - \dot{\mathbf{q}}) \cdot \mathbf{D}. \tag{4.10}$$

Unter Vernachlässigung der Reibung ist dies eine in $\mathbf{q}_a$ lineare Differentialgleichung. Das bedeutet, dass die Positionen der Armelemente $\mathbf{q}_a$ durch lineare Filterung der Kommandos $\mathbf{q}_c$ und der motorseitigen Gelenkwerte $\mathbf{q}$ berechnet werden können. Dabei ist es bei Adaption des Beobachters nicht nötig, die genaue Struktur von Gleichung (4.10) zu kennen. Die vernachlässigte Reibung beschränkt bei dieser Betrachtung die Genauigkeit des Beobachters.

Eine anschaulichere Motivation der in Abb. 4.17 angenommenen Beobachtereingänge ist, dass die Differenz zwischen Motor- und Armelementposition aufgrund von Gleichung (4.7) linear ist. Bei nicht modellierten Verhältnissen im Gelenk, insbesondere bei vorhandener Reibung, wird die Gleichung von der Armseite betrachtet. Das stellungsabhängige Armträgheitsmoment hat neben dem Einfluss auf die Beschleunigung des Armelements auch Rückwirkungen auf die Motorposition. Ein Vergleich der Armelementposition mit einer aufgrund der Kommandos erstellten Vorhersage gibt also Aufschluss über das aktuell wirkende Armträgheitsmoment. Bei Adaption des Beobachters ergibt sich diese Vorhersage implizit. Da in dem Fall das Getriebe von der Armseite betrachtet wird, müssen die Kontaktkräfte extra berücksichtigt werden. Bei Änderung der Nutzlast muss der Beobachter neu adaptiert werden.

Mit den in Abb. 4.17 angenommenen Beobachtereingängen lassen sich also unterschiedliche Beobachter bilden. Eine Adaption wird i. A. nicht den einfacheren Ansatz von Gleichung (4.10) abbilden sondern eine Kombination des antriebs-

und des abtriebsseitigen Ansätzes. Der resultierende Beobachter wird daher von der Konfiguration während der Adaption abhängen, also von Nutzlast und externen Kräften während der Experimente. Für Regelungen mit anderer Konfigurationen ist er nicht verwendbar. Durch Training mit unterschiedlichen Versuchsanordnungen ist es jedoch möglich, eine konfigurationsunabhängige Abbildung zu generieren. Dies ist die Abbildung nach Gleichung (4.10).

4.5 Erwartete Genauigkeit und Resümee

Für das vorgestellte hierarchische System mit 3 Ebenen wurden die beiden oberen Ebenen neu entworfen, während die untere Ebene in Form einer industriellen Steuerung angenommen wurde. Die oberste Ebene beinhaltet eine aufgabenabhängige Bahnplanung, die Robotertrajektorien im Takt der Lageregelung zur Verfügung stellt. Die mittlere Ebene beinhaltet eine prädiktive Vorsteuerung, die die Regelung der Industrierobotersteuerung ergänzt. Sinn dieser Aufteilung ist die strikte Trennung von Bahnplanung und Regelung oberhalb der vorhandenen Schnittstelle der angenommenen Industrierobotersteuerung.

Die Vorsteuerungsebene ist adaptiv konzipiert, wobei die Bahnfehler nach dem Abfahren vorgegebener Trajektorien offline minimiert werden können. Die Adaption selbst erfolgt in drei Schritten. Nach einer Parameteridentifikation des durch die erste Hierarchieebene der Steuerung geregelten Roboters erfolgt eine Trajektorienoptimierung. Sie modifiziert die an der Industrieroboterschnittstelle abgegebenen Bewegungskommandos so, dass die resultierende Bewegung annähernd der Vorgabe entspricht. Im dritten Schritt werden schließlich die Reglerparameter eingestellt. Durch einen geeigneten Regleransatz ist das für eine einzelne Beispieltrajektorie nahezu optimal trainierte Verhalten mit geringen Genauigkeitseinbußen auch auf anderen Trajektorien übertragbar. Durch iteratives Anpassen der Parameter und Überprüfen der verbleibenden Bahnabweichungen wird schließlich eine Bahngenauigkeit erreicht, die durch direkte Berechnung aus dem nur grob identifizierten Modell nicht möglich ist. So ist es auch bei komplexen physikalischen Zusammenhängen ausreichend, wenn das Modell eine linearisierte Abbildung realisiert.

Dabei werden drei Betriebsarten unterschieden. Im einfachsten Fall einer Bahnsteuerung entfällt die dritte Stufe der Adaption. Stattdessen werden die ermittelten Bewegungskommandos lediglich gespeichert und im Anwendungsfall

abgerufen. Dadurch ist nur das Abfahren der Trajektorie möglich, an der die Adaption erfolgte. In den anderen Fällen wird eine Filtergleichung geschätzt und nach der Adaption in einem Vorsteuerungsmodul online auf die gewünschte Bahn angewandt. Dabei ist der Vollständigkeit halber auch ein dritter, aufwendigerer Fall beschrieben, in dem neben den systematischen auch noch stochastische Bahnabweichungen minimiert werden müssen.

Neben der üblichen Annahme des Roboters als Starrkörperstruktur wird auch der Fall elastischer Gelenke berücksichtigt, da Elastizitäten bei den mit der entworfenen Architektur erreichbaren Geschwindigkeiten und Beschleunigungen nicht mehr vernachlässigt werden können.

Die dynamischen Bahnverbesserungen sind abhängig von der Bahngeschwindigkeit und den dabei auftretenden Bahnfehlern. So können geringe Fehler, wie sie auch bei langsamen Bewegungen auftreten, durch Vorsteuerung bei schnellen Bewegungen nicht korrigiert werden. Die Bahngenauigkeit langsamer Bewegungen ergibt also eine obere Schranke für die Verbesserung. Diese Genauigkeit ist schlechter als die Reproduzierbarkeit von Roboterbewegungen, etwa in der Größenordnung, die vom Hersteller als Wiederholgenauigkeit von Positionierbewegungen (Positioniergenauigkeit) angegeben wird, also bei z. B. 0.1 mm. Deutliche Vorteile sind dementsprechend nur dann zu erwarten, wenn bei einer Testbahn ohne besondere Maßnahmen ein wesentlich höherer Bahnfehler beobachtet wird.

Aufgrund der mit größenordnungsmäßig 100 Hz langsamen Abtastung[8] auf der mittleren Hierarchieebene und aufgrund des allgemeinen Ansatzes muss im Gegensatz zu explizit modellbasierten Verfahren damit gerechnet werden, dass die Positioniergenauigkeit zumindest bei problematischen Bewegungen, bei denen z. B. die Bewegungsrichtung eines Gelenks sich ändert, nicht erreicht wird. Ähnliches gilt je nach Implementierung der Vorsteuerung auch für Bahnen, bei denen die Trägheitsmomente zwischen verschiedenen Positionen stark schwanken.

Bei Bahnsteuerungen können allgemein höhere Genauigkeiten erzielt werden. Die Bahngüte wird dabei durch die Repräsentationsfähigkeiten des Modells und ggf. durch Quantisierungs- oder Abtasteffekte eingeschränkt. Die Unterschiede

[8]Die Abtastfrequenz der mittleren Hierachieebene ist durch die Industrieroboterschnittstelle zum Roboter vorgegeben und kann nur in engen Grenzen modifiziert werden. Aber auch unabhängig von dieser Einschränkung sind hohe Abtastfrequenzen mit überproportional hohem Rechenaufwand und dadurch bedingt ggf. mit numerischen Ungenauigkeiten verbunden, sodass derzeit für die mittlere Hierarchiebene von Abtastzeiten im Bereich einer Millisekunde abgeraten wird.

in der Ausführung zwischen Bahnen mit identischen Bewegungsanweisungen (Reproduzierbarkeit des Roboters) beschreiben die höchstens erreichbare Genauigkeit.

Bei der Verallgemeinerung einer Vorsteuerung auf andere (untrainierte) Bahnen kommt es einerseits darauf an, wie stark sich die Bahnen von der Trainingsbahn unterscheiden bzw. wie stark sich die wirksamen Trägheitsmomente unterscheiden. Andererseits wirkt sich nachteilig aus, wenn die Roboterdynamik bei der Trainingsbahn nur unzureichend angeregt wurde. Bei Robotern mit hoch übersetzenden Getrieben sind die Trägheitsmomente der Motoren dominant. Daher wird bei solchen Geräten die an einer „schwierigen“ Bewegung erreichte Bahngenauigkeit auch auf andere Bewegungen übertragbar sein.

Sofern der Einsatz eines externen Messgerätes aufgrund von Getriebe- und Motor-Elastizitäten erforderlich wird, lässt sich die erreichbare Genauigkeit nur schwer abschätzen. Sie wird durch die Genauigkeit des Messsystems und der Kalibrierung begrenzt, wobei das Messrauschen auch auf die Kalibrierung wirkt. Unsicher ist dagegen, ob Eigenschwingungen der Armelemente angeregt werden, die aufgrund der langsamen Abtastung nicht wirksam gedämpft werden können. Selbst wenn dies kein Problem darstellt, so ist doch zu erwarten, dass die Verallgemeinerung auf andere Bahnen nur eingeschränkt möglich ist, da das auf die Getriebeelastizität wirkende Trägheitsmoment das Motorträgheitsmoment nicht enthält, also in jedem Fall positionsabhängig ist.

Bei online sensorgestützt spezifizierten Bahnen ist die Regelgüte sowohl durch die Sensorauflösung und die Signalverarbeitung (Totzeit) als auch durch die Möglichkeiten zur Prädiktion zukünftiger Soll-Bahnpunkte eingeschränkt. Abhängig vom Sensortyp ergeben sich dadurch Nachteile gegenüber fest vorgegebenen Soll-Bahnen, was in Kapitel 6 anhand einiger Beispiele gezeigt wird.

Kapitel 5

Adaptionsvorgang der vorausplanenden Steuerung

In diesem Kapitel wird die vorgeschlagene vorausplanende Robotersteuerung bezüglich ihrer Eigenschaften diskutiert. Dabei wird angenommen, dass die auszuführenden Trajektorien a priori vollständig bekannt seien, so dass es sich um die Ausführung explizit bekannter und zeitinvarianter Bewegungsbahnen handelt. Der Adaptionsvorgang selbst erfolgt off-line anhand einiger Beispieltrajektorien, während die adaptierte Steuerung dann online die erforderlichen Steuerungsanweisungen zur konkreten Ausführung der Trajektorie in Echtzeit generiert.

Es handelt sich um eine Adaption mit Identifikationsmodell. Daher wird zunächst auf die Identifikation eingegangen. Die Struktur des Reglers und die Berechnung der Reglerparameter werden eingehend diskutiert.

5.1 Identifikation von Struktur und Parametern des Roboters mit unterlagerter Regelung

Im Folgenden wird die Identifikation des Roboterstreckenmodells näher betrachtet. Zunächst aber werden die wichtigsten Identifikationsverfahren vorgestellt.

5.1.1 Mögliche Identifikationsverfahren

Man kann Identifikationsverfahren nach der Art der Modellrepräsentation in *parametrische* und *nichtparametrische* Methoden einteilen.

Zunächst seien Parameterschätzverfahren näher betrachtet, die bei Vorgabe einer Modellordnung und -struktur der zu identifizierenden Roboteranordnung automatisch die wichtigsten Modellparameter generieren. Bei diesen Verfahren ist zur Messung eine bekannte Eingangsfunktion nötig, wie z. B. eine Sprungfunktion, die die Strecke ausreichend angeregen kann.

Man geht von einem zeitinvarianten parametrischen Systemmodell aus, das entweder kontinuierlich als inhomogene lineare Differentialgleichung

$$\begin{aligned} y^{(n)}(t) &= b_0 \cdot u(t) + b_1 \cdot \dot{u}(t) + \cdots + b_{n-1} \cdot u^{(n-1)}(t) \\ &- a_0 \cdot y(t) - a_1 \cdot \dot{y}(t) - \cdots - a_{n-1} \cdot y^{(n-1)}(t) \end{aligned} \tag{5.1}$$

oder zeitdiskret als lineare Differenzengleichung

$$\begin{aligned} y(k) &= b_0 \cdot u(k) + b_1 \cdot u(k-1) + \cdots + b_{n-1} \cdot u(k-n+1) \\ &+ a_1 \cdot y(k-1) + a_2 \cdot y(k-2) + \cdots + a_m \cdot y(k-m) \end{aligned} \tag{5.2}$$

formuliert werden kann. Dabei sind u und y die zeitlichen Funktionen der Ein- und Ausgabegrößen des betrachteten Systems bzw. die messbaren Werte zu den jeweiligen Abtastzeitpunkten $k, k-1, \cdots$. a_i und b_i sind die zu identifizierenden Parameter. Sie unterscheiden sich in beiden Systemrepräsentationen.

Im Folgenden werden diskrete Darstellungen genutzt, da sie einfacher handhabbar und bei linearen Systemen gleichwertig zu kontinuierlichen Repräsentationen sind. Kontinuierliche Repräsentationen erlauben bei nichtlinearen Ansätzen eine höhere Genauigkeit, da physikalische Effekte wie z. B. Reibung explizit modelliert werden können. Im Rahmen der mehrstufigen Adaptionsstruktur dieser Arbeit ist dieser Aufwand aber nicht erforderlich.

Üblicherweise werden die Variablen zusammengefasst zu einem Messvektor

$$\psi(k) = (u(k), u(k-1), \cdots, u(k-n+1), y(k-1), y(k-2), \cdots, y(k-m))^T \quad (5.3)$$

und einem Parametervektor

$$\theta = (b_0, b_1, \cdots, b_{n-1}, a_1, a_2, \cdots, a_m)^T, \quad (5.4)$$

woraus sich

$$y(k) = \psi^T(k) \cdot \theta \quad (5.5)$$

als mathematisches Systemmodell ergibt. Dabei ist der Vektor θ aufgrund der gesamten Trajektorie mit N Abtastschritten zu identifizieren.

Aufgrund der numerischen Stabilität wird meist ein rekursives Schätzverfahren eingesetzt, bei dem die Parameter vom Anfangsvektor $\hat{\theta}(0)$, meist $\hat{\theta}(0) = \mathbf{0}$, schrittweise verbessert werden. Der auf Basis der Daten bis zum Zeitpunkt $k+1$ identifizierte Parametervektor wird als Schätzvektor $\hat{\theta}(k+1)$ bezeichnet. Er ergibt sich aus folgendem rekursiven Least-Square (RLS) Algorithmus (siehe z. B. [109]):

$$\begin{aligned} \hat{\theta}(k+1) &= \hat{\theta}(k) + \mathbf{P}^*(k+1) \cdot \psi(k+1) \\ &\cdot \quad (\psi^T(k+1) \cdot \mathbf{P}^*(k+1) \cdot \psi(k+1) + 1)^{-1} \\ &\cdot \quad (y(k+1) - \psi^T(k+1) \cdot \hat{\theta}(k)) \end{aligned} \quad (5.6)$$

$$\begin{aligned} \mathbf{P}(k+1) &= \mathbf{P}^*(k+1) - \mathbf{P}^*(k+1) \cdot \psi(k+1) \cdot \psi^T(k+1) \cdot \mathbf{P}^*(k+1) \\ &\cdot \quad (\psi(k+1)^T \cdot \mathbf{P}^*(k+1) \cdot \psi(k+1) + 1)^{-1}. \end{aligned} \quad (5.7)$$

Dabei sind $\psi(k+1)$ und $y(k+1)$ der oben definierte Messvektor und die skalare Messgröße[1]. $\mathbf{P}$ und $\mathbf{P}^*$ sind symmetrische Matrizen ohne physikalische Bedeutung. Sie stellen auch keine Wahrscheinlichkeiten dar. Im Fall des einfachen RLS-Algorithmus ist

[1]Bei dieser Darstellung wird von einer skalaren Messgleichung (5.5) ausgegangen. Eine Erweiterung auf den vektoriellen Fall ist leicht möglich, wird im Rahmen dieser Arbeit aber nicht verwendet.

$$\mathbf{P}^*(k+1) = \mathbf{P}(k). \tag{5.8}$$

$\mathbf{P}$ wird meist durch eine Diagonalmatrix $\mathbf{P}(0)$ mit „großen“ Werten initialisiert und dann in jedem Schritt modifiziert.

Die Gleichungen (5.6) und (5.7) lassen sich wie folgt interpretieren: Gleichung (5.6) modifiziert den zuletzt identifizierten Parametervektor $\hat{\theta}(k)$ aufgrund neuer Messungen. Dabei wird zunächst ein Messwert $\hat{y}(k+1) = \psi^T(k+1) \cdot \hat{\theta}(k)$ vorhergesagt, dessen Differenz zum tatsächlichen Messwert $y(k+1)$, gewichtet mit $\mathbf{P}^*(k+1) \cdot \psi(k+1) \cdot (\psi^T(k+1) \cdot \mathbf{P}^*(k+1) \cdot \psi(k+1) + 1)^{-1}$, die Änderung des Parametervektors ergibt. Die Gewichtung selbst wird in Gleichung (5.7) modifiziert, so dass der Vorhersagefehler des Messwerts zu Beginn stark, später aber gefiltert auf den Parametervektor einwirkt. Insgesamt ergibt sich durch den Algorithmus eine Minimierung des quadratischen Vorhersagefehlers.

Der Algorithmus weist Ähnlichkeiten zum Kalman Filter [199] auf (siehe auch Anhang A). Neben dem Algorithmus nach Gleichung (5.6) und (5.7) gibt es verschiedene Varianten, z. B. das gewichtete rekursive Least-Square (WRLS) Verfahren, bei dem die P-Matrix zwischen den Abtastschritten derart angepasst wird, dass die unmittelbar zurückliegenden Schritte stärker gewichtet werden. Üblich ist

$$\mathbf{P}^*(k+1) = \mathbf{P}(k)/\lambda \tag{5.9}$$

mit einem Skalar $\lambda < 1$, wie in [109]. Dadurch wird die Norm der P-Matrix vor der Berücksichtigung der neuesten Messwerte erhöht, wodurch diese Messwerte größeren Einfluss bekommen. Die Wahl von λ ist durch die Bedingung eingeschränkt, dass die Iteration aus Gleichung (5.7) und (5.9) nicht divergieren darf.

Neben der rekursiven Fassung gibt es auch eine nichtrekursive Form des Algorithmus, die die Daten aus allen N Abtastzeitpunkten zusammenfasst:

$$\mathbf{\Psi} = (\psi(1), \cdots, \psi(N)) \tag{5.10}$$

$$\mathbf{y} = (y(1), \cdots, y(N))^T \tag{5.11}$$

Die Gleichungen

$$\hat{\theta}(N) = (\mathbf{\Psi} \cdot \mathbf{\Psi}^T + \mathbf{P}_0^{-1})^{-1} \cdot \mathbf{\Psi} \cdot \mathbf{y} \tag{5.12}$$

$$\mathbf{P}(N) = (\mathbf{\Psi} \cdot \mathbf{\Psi}^T + \mathbf{P}_0^{-1})^{-1} \tag{5.13}$$

sind äquivalent zum einfachen RLS-Algorithmus. Für $\mathbf{P}_0^{-1} \approx \mathbf{0}$ und Matrizen $\mathbf{\Psi}$ mit Rang n lässt sich Gleichung (5.12) direkt aus dem Systemmodell (5.5) herleiten.

Weitere Identifikationsansätze sind in [109] zu finden.

5.1.2 Wahl der Modellstruktur

Die Identifikationsaufgabe zur Realisierung der adaptiven vorausplanenden Steuerung unterscheidet sich von der modellbasierten Identifikationen von Robotern mit Momentenschnittstelle (siehe z. B. [192, 250]). Durch die Nutzung von Positionsschnittstellen wirkt ein durch eine Kaskadenregelung geregelter Roboter als zu identifizierende Strecke und nicht nur das mechanische System selbst.

Dabei ist die genaue Struktur einer solchen Regelung von Hersteller zu Hersteller unterschiedlich und oft im Detail unbekannt. Deshalb ist es nur schwer möglich, analytisch entworfene Robotermodellstrukturen mit den Strukturen der Regelungen zu Gesamtmodellen der geregelten Roboter zu kombinieren. Die Vorgabe von Strukturen der zu identifizierenden Systeme wird dabei nicht notwendigerweise benötigt.

Auf der anderen Seite kann eine unvollständige Nachbildung von realen Strukturen von zu identifizierenden Systemen dazu führen, dass die Modelle die realen Systeme nicht repräsentieren können. Theoretisch ist also ein nichtparametrisches Modell zur Identifikation dynamischer Modelle von Robotern mit unbekannter Struktur am besten geeignet, also Modelle, bei denen keine Strukturen vorgegeben sind, wie beispielsweise bei Neuronalen Netzen.

Zur Unterstützung der Optimierungsstufe eines Adaptionsverfahrens ist es aber von Vorteil, wenn die Modelle parametrisch bezüglich der zu verfahrenden Tra-

jektorie vorliegen, da dann für die Optimierungsfunktion ein Gradientenverfahren gewählt werden kann. Bei nichtparametrischen Modellen werden stattdessen Suchverfahren eingesetzt (siehe z. B. bei Ersü oder Tolle [77, 238] oder Ozaki et al. [194]), die unterschiedliche Bewegungsanweisungen generieren, deren Auswirkungen erst durch Aufruf des Robotermodells ermittelt werden. Weiter besteht bei nichtlinearen Modellen adaptiver Systeme das Problem, die Ausführung unterschiedlicher Bewegungsbefehle zu vergleichen, was zu fehlerhaften Schlüssen führen kann. Daher ist es sinnvoll, für die zu identifizierenden Modelle im Gegensatz zu den Reglern eine feste Struktur vorzugeben. Möglich sind auch Modelle, die bei der Gewichtung unterschiedlicher Bewegungsanweisungen feste Strukturen ansetzen, die aber keine Einschränkung bzgl. der Parametervariation in Abhängigkeit des Roboterzustands aufweisen. Zu dieser Klasse gehören Überlagerungen aus parametrischen und nichtparametrischen Modellen wie z. B. bei ME9 [148].

Gesucht ist also ein allgemeines Modell, das auf der Basis eines vereinfachten Streckenansatzes identifiziert werden kann. Die Schätzung der Parameter eines linearisierten Modells ist dabei einfacher als die Identifikation eines aufgrund von Strukturkenntnissen abgeleiteten nichtlinearen Modells. Es zeigt sich, dass dieses Modell bei vielen Robotern nahezu unabhängig von den zur Linearisierung gewählten Arbeitspunkten ist.

In zeitkontinuierlicher Darstellung ergibt sich die Übertragungsfunktion der zu identifizierenden Strecke zu:

$$\begin{aligned} Y^{(n)}(t) &= b_0 \cdot U(t) + b_1 \cdot \dot{U}(t) + \cdots + b_{n-1} \cdot U^{(n-1)}(t) \\ &- a_0 \cdot Y(t) - a_1 \cdot \dot{Y}(t) - \cdots - a_{n-1} \cdot Y^{(n-1)}(t) \end{aligned} \tag{5.14}$$

mit $U(t) = u(t) - u_0$ und $Y(t) = y(t) - y_0$. Gleichung (5.14) ist allgemeiner als Gleichung (5.1), da dort angenommen wird, dass der Nullpunkt der Gelenkwinkelmessung bei $u = y = 0$ einen Gleichgewichtspunkt darstellt. In Gleichung (5.14) ist dagegen der zur Linearisierung gewählte Gleichgewichtspunkt durch u_0 und y_0 frei.

Zeitkontinuierliche Darstellungen sind dann sinnvoll, wenn physikalische Abhängigkeiten beschrieben werden sollen, die durch Differentialgleichungen gegeben sind. Dies gilt z. B. für dynamische Robotermodelle, die die Struktur der mechanischen Zusammenhänge formelmäßig widerspiegeln. Analytische Mo-

delle verlieren an Aussagekraft, wenn sie zeitdiskret repräsentiert werden, da nichtlineare Effekte wie der Übergang aus der Haftreibung in Gleitreibung nicht immer zu den Abtastzeitpunkten erfolgen.

Linearisierte Gleichungen wie Gleichung (5.14) lassen sich dagegen korrekt diskretisieren. Zeitdiskrete Systemrepräsentationen wie

$$Y(k+1) = b_1 \cdot U(k) + b_2 \cdot U(k-1) + \cdots + a_1 \cdot Y(k) + a_2 \cdot Y(k-1) + \cdots \quad (5.15)$$

sind bei Abtastregelungen von Vorteil, wenn beispielsweise die zur Berechnung von Gleichung (5.14) benötigten Ableitungen der Signale nicht verfügbar sind. Bei zeitkontinuierlichen Modellen ist somit eine Filterung der Ein- und Ausgangssignale der Strecke wie z. B. nach Prüfer et al. [205, 206] nötig.

Die Übertragungsfunktion nach Gleichung (5.15) wird bei adaptiven oder lernenden Regelungssystemen häufig angewandt (z. B. in [140]). Dagegen soll im Folgenden das zu identifizierende Robotermodell als Gewichtsfunktion (Impulsantwort, auch FIR = finite impulse response)[2] repräsentiert werden.

$$y(k+1) - y_0 = \hat{g}_1 \cdot (u(k) - u_0) + \hat{g}_2 \cdot (u(k-1) - u_0) + \hat{g}_3 \cdot (u(k-2) - u_0) + \cdots \quad (5.16)$$

Diese Darstellung lässt sich leichter auswerten als Gleichung (5.15).

In Gleichung (5.16) sind die vorgegebenen Gelenkwinkel mit u, die gemessenen Winkel mit y bezeichnet. Aufgrund der unterlagerten Regelung kann jeder Arbeitspunkt statisch definiert werden. Als Gleichgewichtspunkt, um den linearisiert wird, kann also jeder Punkt $y_0 = u_0$ vorgegeben werden. Es ist dabei durchaus zulässig, den Linearisierungspunkt durch $u_0 = u(k)$ von Schritt zu Schritt zu verändern.

Die Sprungantwort des Roboters auf eine Bewegungsanweisung konvergiert ohne bleibende Regelabweichung gegen den gewünschten Sollwert. Bei einem Einheitssprung ist der Endwert also 1. Dieser Endwert entspricht der mit der Abtastzeit normierten Fläche unter der Impulsantwort. Es gilt also $\sum_{i=1}^{\infty} g_i = 1$.

[2] Ähnlich wie die Sprungantwort die Reaktion des Systems auf einen Einheitssprung darstellt beschreibt die Gewichtsfunktion die Reaktion auf einen Impuls. Bei zeitdiskreten Systemen ist der Impuls auf die Höhe 1 oder $1/T_0$ normiert. Die Gewichtsfunktion ist die zeitliche Ableitung der Sprungantwort.

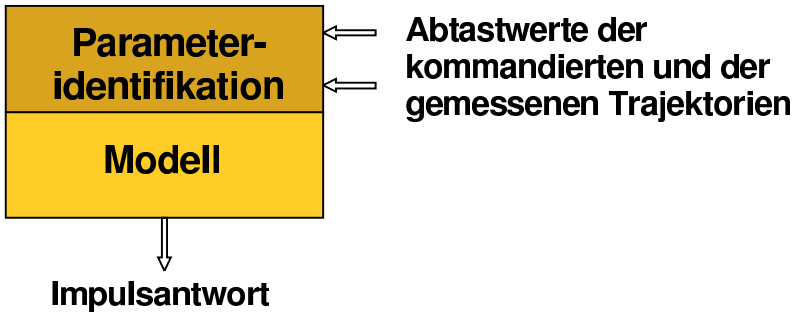


Abbildung 5.1: Schnittstellen der Identifikation des Modells

Auch daran kann man sehen, dass $y_0 = u_0$ beliebig gewählt werden kann, da es sich bei $\sum_{i=1}^{\infty} \hat{g}_i \approx 1$ in Gleichung (5.16) praktisch nicht auswirkt. Die Wahl von $u_0 = u(k)$ ist dabei numerisch günstig. Große Differenzen zwischen u_0 und $u(k)$ bedeuten dagegen eine hohe Empfindlichkeit der durch das Modell vorausgesagten $y(k+1)$ auf Schätzfehler der $\hat{g}_i$.

Die Struktur der Parameteridentifikation und des Modells können Abb. 5.1 entnommen werden.

Abb. 5.2 zeigt typische Verläufe für die identifizierte Gewichtsfunktion. Daran kann man zwei Eigenschaften ablesen: Zum Einen erfolgt die Impulsantwort gegenüber dem Impuls um eine Totzeit von $n_t = 3$ Schritten verzögert. Ursache hierfür sind Rechen- und Datenübertragungszeiten in dem Rechnersystem. Darüber hinaus klingt die Gewichtsfunktion, wie alle Gewichtsfunktionen von stabilen Systemen, schnell ab, so dass nur wenige Elemente der Reihenentwick-

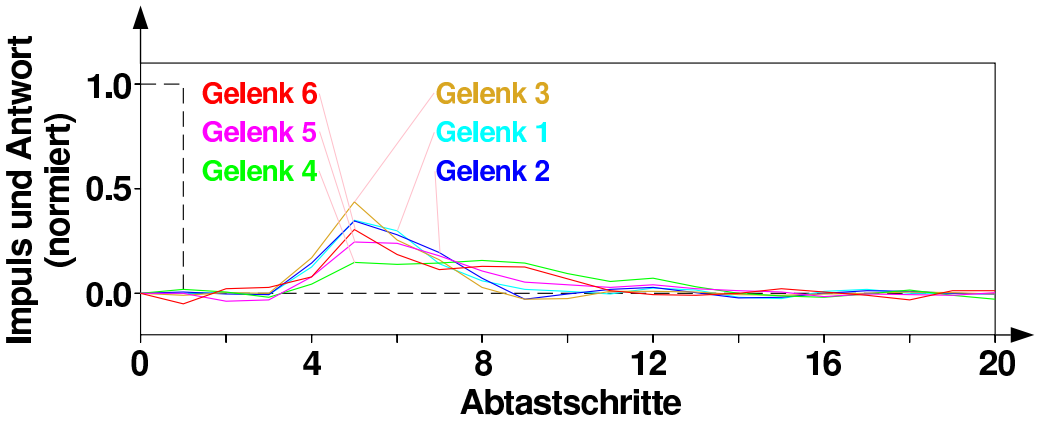


Abbildung 5.2: Diskreter Impuls (gestrichelt) und unterschiedliche, geschätzte Impulsantworten der einzelnen Achsen, berechnet am Beispiel eines Manutec r2 Systems

lung bestimmt werden müssen.

Damit lässt sich Gleichung (5.16) modifizieren:

$$\begin{aligned} y(k+1) &= u(k) + \hat{g}_1 \cdot (u(k-n_t) - u(k)) + \hat{g}_2 \cdot (u(k-n_t-1) - u(k)) \\ &+ \hat{g}_3 \cdot (u(k-n_t-2) - u(k)) + \cdots \\ &\qquad + \hat{g}_{n_m} \cdot (u(k-n_t-n_m+1) - u(k)) \end{aligned} \tag{5.17}$$

n_m bezeichnet dabei die Zahl der identifizierten Parameter $\hat{g}_i$ des Modells.

5.1.3 Identifikation der Parameter

Die Aufgabe der Parameteridentifikation ist es, die Parameter $\hat{g}_i$ der Gewichtsfunktion zu schätzen. Anstelle des Least-Square-Algorithmus nach den Gleichungen (5.6) und (5.7) wird ein rekursives Kalman Filter[3] (siehe Anhang A.1) vorgeschlagen, da dabei im Vergleich zu anderen Least-Square-Verfahren einerseits die Parameter am besten eingestellt werden können und da andererseits die P-Matrix die Kovarianz der Schätzfehler der Parameter angibt. Außerdem erleichtert die Verwendung eines Kalman Filters die Anpassung der Parameter.

Als Parameter des Kalman Filters sind die Varianz der angenommenen Störung σ^2 und der Anfangswert der Kovarianzmatrix (quadratische Erwartungswerte der Schätzwerte) $\mathbf{P}_0$ vorgegeben, wobei eine grobe Schätzung durchaus ausreicht. So bewirken um ein oder zwei Zehnerpotenzen zu hohe oder zu niedrige Vorgaben fast keine Unterschiede für das Modell.

Wichtiger ist allerdings die Robustheit der Parameterschätzung gegenüber Störungen der Messwerte oder numerischen Ungenauigkeiten. Gerade dies ist aber die Stärke von Schätzverfahren. Ein weiteres Argument für die Wahl des Kalman Filters ist in seiner geringen Empfindlichkeit gegenüber unmodellierten Effekten wie Quantisierung der Messsignale oder Kopplungen zwischen den Gelenken zu sehen.

Mit den gewählten Parametern des Kalman Filters kann eine vorläufige Iden-

[3]Für das Kalman Filters wird nur eine Messgleichung betrachtet, nicht aber eine Dynamikgleichung (Systemgleichung). Dadurch bildet das rekursive Kalman Filter nicht, wie sonst üblich, die Dynamik der Regelstrecke ab. Die somit verwendete Fassung des Kalman Filters ist dem oben erläuterten Least-Square-Algorithmus sehr ähnlich.

tifikation der Modellparameter durchgeführt werden. So zeigen mit $n_t = 0$ und $n_m = 20$ ermittelte Gewichtsfunktionen (Abb. 5.2) für die einzelnen Achsen eines Manutec r2 Roboters, dass bei diesem System eine Totzeit von 3 Abtastzeitintervallen aufgrund der Signalverarbeitung angesetzt werden kann. Ab dem 4. Abtastschritt sind die identifizierten Parameter der Gewichtsfunktion deutlich von null verschieden.

Einige Abtastschritte später klingen die Werte wieder ab. Bei manchen zu identifizierenden Achsen reicht es, den Verlauf der Gewichtsfunktion nur bis zu Abtastschritt 8 zu betrachten, also über 5 Abtastzeitintervalle hinweg. Bei ungenauen Identifikationen wie z. B. den Handachsen werden dagegen bis zu 9 signifikante Schätzwerte $\hat{g}_i$ identifiziert.

Aus diesem Grund wird als Strukturparameter für das zu identifizierende Modell bei dem betracheten Manutec r2 Roboter neben $n_t = 3$ noch $n_g = 7$ definiert. Dabei ist n_g die zur Optimierung der Bewegungsanweisungen verwendete Zahl der Elemente der Gewichtsfunktion, die insgesamt n_m Elemente enthält. n_m wird deutlich größer als n_g gewählt, z. B. $n_m = 14$, um abschätzen zu können, welcher Fehler sich durch den Abbruch der Reihenentwicklung nach Abtastschritt $n_t + n_g$ ergibt (siehe Abb. 5.2).

Wenn keine aussagekräftigen Experimente gemacht werden können, ist es sinnvoll, bei unbekannten Roboterstrukturen sicherheitshalber einen kleineren Wert für die Totzeit n_t und einen größeren Wert für n_m+n_t und für n_g+n_t zu wählen.

Die meisten Identifikationsverfahren verwenden ein spezielles Anregungssignal, wie z. B. eine Sprungfunktion oder eine für die Identifikation optimierte Trajektorie nach Swevers et al. [234]. Solche Vorgaben sind bei dem hier verwendeten Schätzverfahren nicht zwingend notwendig. Stattdessen kann prinzipiell eine Trajektorie gewählt werden, die der Aufgabe des Roboters nach der Adaptionsphase entspricht. Eine robustere Schätzung erhält man jedoch, wenn man einer glatten Folge von Soll-Bahnpositionen zusätzlich eine mittelwertfreie stochastische Anregung kleiner Amplitude überlagert, aufgrund derer die Reaktion der Regelstrecke auch für höhere Frequenzen bestimmt werden kann.

Vorgeschlagen wird eine normalverteilte Anregung, die zu einer schnellen Abnahme der Schätzfehler führt. Diese Anregung ist einfacher als das z. B. bei Ersü und Tolle [78] verwendete aktive Lernen (Lernen durch Experimentieren), das die Kommandos gezielt modifiziert, um das (nichtlineare) Modell zu verbessern. Bei den Experimenten in Kapitel 6 zeigt sich, dass eine stochastische

Anregung ausreicht und gleichzeitig kaum zu Abweichungen von der ursprünglich geplanten Bahn führt.

Als Strategie für die Identifikation der Modellparameter erweist es sich als sinnvoll, die vorgegebenen Soll-Bahnen *langsam* und ohne Kraftkontakte abzufahren. Dadurch wird das zu identifizierende Modell wenig gestört. Mit den aufgenommenen Daten dieser Trajektorien kann dann offline das Modell eines geregelten Roboters abgeleitet werden. Aufgrund des linearen Ansatzes ist die Aufnahme weiterer Daten zur Verbesserung des Modells nicht nötig.

5.2 Schätzung der optimalen Stellgrößen

Nach der Identifikation des Modells des geregelten Roboters kann dieses im Prinzip angewandt werden, um die optimalen Stellgrößen für vorgegebene Trajektorien zu bestimmen. Die Schnittstellen zu der Optimierungsfunktion sind in Abb. 5.3 noch einmal aufgeführt.

Die vorgegebenen Trajektorien werden unter Nutzung der Modellinformation a posteriori derart modifiziert, dass die gemessenen Regeldifferenzen unter der Annahme von Störungen beim nochmaligen Abfahren der Trajektorie minimiert werden.

Dazu betrachtet man die Modellgleichung (5.17) mit n_g anstelle von n_m zunächst zur Beschreibung der gemessenen Trajektorie

$$y(k+1) = u(k) + \sum_{i=1}^{n_g} \hat{g}_i \cdot (u(k - n_t - i + 1) - u(k)) + \text{Störung} \qquad (5.18)$$

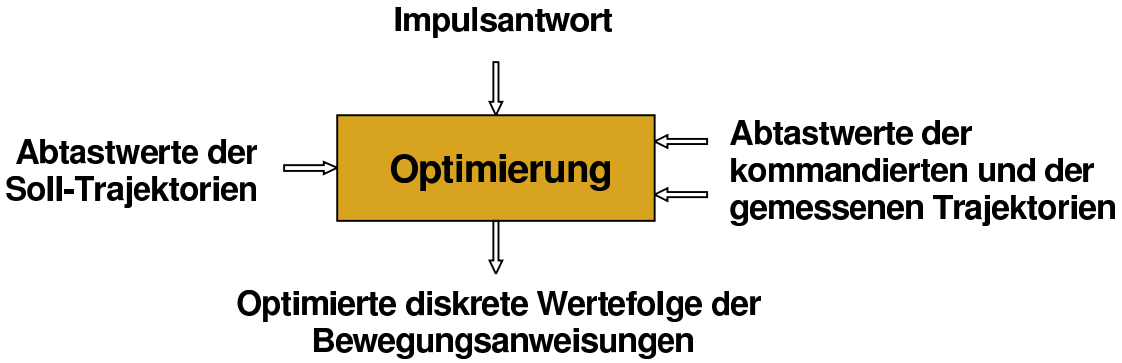


Abbildung 5.3: Schnittstellen der Optimierung der Bewegungsanweisungen

und dann noch einmal zur Beschreibung der gewünschten Bewegung

$$w(k+1) = u(k) + \sum_{i=1}^{n_g} \hat{g}_i \cdot (u_{neu}(k - n_t - i + 1) - u(k)) + \text{Störung}. \qquad (5.19)$$

Die Differenz beider Gleichungen ist von der nicht direkt messbaren Störung unabhägig. Damit ergibt sich eine Vorschrift für die modifizierten Bewegungsanweisungen.

$$w(k+1) - y(k+1) = \sum_{i=1}^{n_g} \hat{g}_i \cdot (u_{neu}(k - n_t - i + 1) - u(k - n_t - i + 1)) \quad (5.20)$$

Um diese rekursive Gleichung nach den Abtastwerten der Stellgröße $u_{neu}(k)$ auflösen zu können, müssen mehrere Zeitpunkte betrachtet werden. Hierzu seien einige erklärende Bemerkungen aufgeführt:

Es wird vorausgesetzt, dass die betrachtete Regelungsstrecke stabil ist. Das ist bei Robotern im Falle einer Positionsschnittstelle gegeben. Bei stabilen Systemen kann die Reihenentwicklung der Impulsantwort, wie im Abschnitt 5.1 gezeigt, nach einigen Elementen abgebrochen werden.

Dagegen ist ein ungeregelter Roboter, also ein System, bei dem direkt die Gelenkmomente bzw. die Gelenkströme als Stellgröße betrachtet werden, eine instabile Regelstrecke. Bei instabilen Systemen kann die Reihenentwicklung der Gewichtsfunktion nicht abgebrochen werden, da die Elemente nicht gegen null konvergieren. Man müsste also eine Gewichtsfunktion ansetzen, deren Zahl an Elementen der Gesamtzahl an Abtastschritten innerhalb einer Trajektorie entspricht ($n_g = N$), was einerseits den Aufwand enorm erhöhen und andererseits zu Konvergenzproblemen führen würde.

Eine andere Betrachtung zeigt, dass bei instabilen Systemen eine Stellgröße zum Zeitpunkt k_1, die den Zustand aus einem (instabilen) Gleichgewichtspunkt auslenkt, wegen der aufklingenden Gewichtsfunktion größere Auswirkungen auf die Zustände an weit in der Zukunft liegenden Zeitpunkten $k_3 \gg k_1$ hat als auf die nächsten Abtastschritte $k_2 = k_1 + 1, k_1 + 2, \cdots$. Die Kompensation einer Regelabweichung zum Zeitpunkt k_3 erfolgt also am einfachsten durch Korrektur der Bewegungsanweisungen von weit zurückliegenden Zeitpunkten $k_1 \ll k_3$.

Dementsprechend korrigiert Gleichung (5.20) aufgrund von hohen Gewichtungen $\hat{g}_i$ insbesondere die ersten Bewegungsanweisungen einer Trajektorie. Unterschiedliche Regelabweichungen in unterschiedlichen Bereichen einer Trajektorie werden dadurch nicht unabhägig voneinander minimiert. Die vorgeschlagene Adaptionsstruktur ist also nicht geeignet, instabile Regelstrecken wie einen ungeregelten Roboter zu stabilisieren.

Zur Anwendung eines stabilen Streckenmodells muss man den Zeithorizont festlegen. Die Optimierung der Bewegungsanweisungen konvergiert in der Regel nur dann, wenn die Strecke über eine genügend große Zeit beobachtet wird. Oben wurde schon erwähnt, dass eine 1-Schritt-Prädiktion, also die schrittweise Auflösung von Gleichung (5.20) mit den Anfangsbedingungen $w(k) = y(k) \quad \forall k \leq 0$ zu

$$\begin{aligned}
u_{neu}(0-n_t) &= u(0-n_t) + \frac{w(1)-y(1)}{\hat{g}_1} \\
u_{neu}(1-n_t) &= u(1-n_t) + \frac{w(2)-y(2)}{\hat{g}_1} \\
&- \frac{\hat{g}_2 \cdot (u_{neu}(0-n_t) - u(0-n_t))}{\hat{g}_1} \\
u_{neu}(2-n_t) &= u(2-n_t) + \frac{w(3)-y(3)}{\hat{g}_1} \\
&- \frac{\hat{g}_2 \cdot (u_{neu}(1-n_t) - u(1-n_t))}{\hat{g}_1} \\
&- \frac{\hat{g}_3 \cdot (u_{neu}(0-n_t) - u(0-n_t))}{\hat{g}_1} \\
&\ldots
\end{aligned} \tag{5.21}$$

und damit zu stark schwankenden Stellgrößen und somit nicht zum globalen Optimum führt. Bei den Gewichtsfunktionen nach Abb. 5.2, bei denen das erste Element nicht gleichzeitig das größte ist, konvergiert die 1-Schritt-Prädiktion nicht.

Darüber hinaus ist nicht jedes ausgeführte Trajektorienelement nach der Optimierung durch Gleichung (5.20) auch Element der gesuchten optimalen Stellfolge. Man muss zwei Fälle unterscheiden:

Bei der Ausführung einer konstanten Trajektorie ohne stochastische Störungen

kann auf eine parallele Messwertrückführung des Roboterzustands verzichtet werden. Dies ist in den Abbildungen 4.8 und 4.9 dargestellt. Danach müssen alle optimierten Stellgrößen von einem Zustand *auf* der optimalen Trajektorie zum nächsten optimalen Steuerungsschritt führen. Dieser Fall wird im nächsten Abschnitt behandelt.

Bei stochastischen Störungen ist dagegen eine Messwertrückführung des Roboterzustands nötig, vergl. Abb. 4.10. Für Situationen dieser Art sollen in der dritten Stufe der Adaption aufgrund der optimierten Stellfolge aus der zweiten Stufe die Reglerparameter derart gefunden werden, dass sie den geregelten Roboter von beliebigen Zuständen auf die Soll-Trajektorie zurücksteuern. Zum Training der Messwertrückführung in der unteren Stufe der Adaption ist es also sinnvoll, von Teiltrajektorien auszugehen, die jeweils von einem *gemessenen* Zustand $(y(k), u(k), u(k-1), \cdots)$ auf die Soll-Trajektorie $w(k+i)$ führen. Dabei können die *gemessenen* Zustände *auf* oder *neben* der gewünschten Trajektorie liegen.

5.2.1 Auswertung der Gewichtsfunktion zur Bahnsteuerung

Im Folgenden wird auf den Fall einer Bahnsteuerung oder Vorsteuerung eingegangen, in dem nur das Verfolgen einer vorgegebenen Trajektorie trainiert wird. Dabei werden alle Abtastpunkte der Trajektorie betrachtet. Der Zeithorizont entspricht also der Länge der Trajektorie.

Zur Optimierung einer einzigen vorgegebenen Trajektorie erhält Gleichung (5.20) unter Verwendung von $e = w - y$ und von $\Delta u = u_{neu} - u$ folgende Form:

$$e(k+1) = \sum_{i=1}^{n_g} \hat{g}_i \cdot \Delta u(k - n_t - i + 1) \tag{5.22}$$

Für die gesamte Trajektorie ergibt diese Schreibweise

$$\begin{bmatrix} \hat{g}_1 & & & & \\ \cdots & \cdots & & & \\ \hat{g}_{n_g} & & \cdots & & \\ & \cdots & & \cdots & \\ & & \hat{g}_{n_g} & \cdots & \hat{g}_1 \end{bmatrix} \cdot \begin{bmatrix} \Delta u(0) \\ \cdots \\ \cdots \\ \cdots \\ \Delta u(N-1) \end{bmatrix} = \begin{bmatrix} e(n_t+1) \\ \cdots \\ \cdots \\ \cdots \\ e(n_t+N) \end{bmatrix} \tag{5.23}$$

Die Lösung dieses Gleichungssystems ist direkt möglich, kann aber schrittweise zu stark unterschiedlichen Stellgrößen führen, wenn bei der 1-Schritt-Prädiktion $\Delta u(0)$ so gewählt wird, dass $y(n_t+1)$ dem Sollwert entspricht, $\Delta u(1)$ den Fehler $e(n_t+2)$ minimiert und so weiter.

Zur Reduktion der Empfindlichkeit der Δu bezüglich Messstörungen in e empfiehlt sich stattdessen eine Kalman Filterung[4], ähnlich wie bei dem modellbasierten Parameteridentifikationsansatz aus Abschnitt 5.1. Der Aufwand der Kalman Filterung steigt quadratisch[5] mit der Zahl der Unbekannten, wodurch die Schätzung sehr rechenintensiv ist. Deswegen wurde eine Variante eines inversen Kalman Algorithmus entwickelt (siehe Anhang A.2), deren Aufwand lediglich linear von der Zahl der von null verschiedenen Elemente der Koeffizientenmatrix abhängt. Da die Gewichtsfunktionen, wie oben gezeigt, aufgrund der Stabilität der Regelungsstrecke nach n_g Schritten abgebrochen werden dürfen, ist der Aufwand der Schätzung lediglich linear zu der Zahl der Abtastschritte.

Im Gegensatz zu dem modellbasierten Parameteridentifikationsansatz in Abschnitt 5.1 wird der Kalman Algorithmus hier untypisch verwendet. Das rekursive Kalman Filter wurde zur Filterung gestörter Systeme entwickelt, deren Zustand oder Parameter mit fortschreitender Zeit immer genauer bestimmt werden. Dagegen wird es in diesem speziellen Fall als Schätzverfahren zur robusten Bestimmung der $\Delta u(k)$ in Gleichung (5.23) genutzt. Die Abweichungen von der gewünschten Trajektorie werden als stochastisches Messrauschen angenommen, obwohl sie eigentlich eine deterministische Funktion bilden.

Das bedeutet, dass bei inkrementeller Verbesserung, also bei nochmaligem Abfahren einer Trajektorie unter Verwendung der aktuell bestimmten Bewegungsanweisungen die ursprüngliche Filterung nicht fortgesetzt werden kann. Statt-

[4] Auch hier wird nur eine Messgleichung betrachtet, sodass des Kalman Filters als spezielle Form eines Least-Square-Algorithmusses gesehen wird, ohne Berücksichtigung einer Dynamikgleichung (Systemgleichung), die ein zu filterndes System abbildet.

[5] Der Aufwand des Kalman Filters steigt mit der dritten Potenz der Zahl der Schätzwerte. Ein quadratischer Aufwand ergibt sich nur, wenn die in der Prädiktionsgleichung vorgesehene Matrizenmultiplikation wegen eines einfachen Systemmodells mit $\mathbf{\Phi} = \mathbf{I}$ entfällt. Dies ist im Anhang A näher erläutert.

dessen handelt es sich um ein neues Schätzproblem. Das bedeutet, dass die Kovarianzmatrix nach Bestimmung der korrigierten Stellgrößenfolge nicht weiter verwendet werden kann. Stattdessen werden bei nochmaligem Abfahren der Trajektorie zur erneuten Optimierung der Bewegungsanweisungen die Kovarianzmatrix und die Schätzwerte $\Delta u(k)$ neu initialisert, um die zuletzt aufgetretenen Positionsfehler in gleicher Weise wie beim ersten Durchlauf zu reduzieren.[6]

Auf diese Art ist eine iterative Korrektur der Stellfolge möglich. Die Iteration hat den Vorteil, dass nach der Korrektur der Stellfolge unerwartet aufgetretene Regeldifferenzen auch korrigiert werden können. Solche neu hinzukommenden Fehler können auftreten, da das Modell nur eine linearisierte Näherung der nichtlinearen Strecke ist. Ein Kalman Filter liefert Schätzwerte für ein gestörtes System bei hinreichend bekanntem Modell. D. h. bei mehrmaligem Abfahren einer Trajektorie werden die Messungen gleichgewichtig gefiltert. Dagegen sollen bei Annahme eines ungenauen Modells nur die Messwerte der jeweils letzten Iteration berücksichtigt werden. Das erfordert die Initialisierung des Kalman Filters zu Beginn jeder Iteration. Außerdem ist die Initialisierung wichtig bei der Adaption der Vorsteuerung an unterschiedlichen Trajektorien oder bei Optimierung von Bewegungsabläufen mit stochastischen Störungen.

Die Parameter $\mathbf{P}_0$ und σ^2 der Kalman Schätzung, also die quadratischen Erwartungswerte von Δu und der Messstörung, können aufgrund des Modells adaptiert werden. So lässt sich durch Vergleich der mit Hilfe des a priori bekannten Modells vorhergesagten Werte mit den gemessenen Ist-Werten eine Analyse der wirkenden Störungen und der Modellgenauigkeit durchführen.

Die Bestimmung der Δu in Gleichung (5.23) wird als Optimierungsaufgabe bezeichnet. Durch die Verwendung des Kalman Algorithmus kann aber explizit kein Gütekriterium vorgegeben werden. Der Kalman Algorithmus minimiert ähnlich wie Least-Square Verfahren den quadratischen Gleichungsfehler, hier die quadratischen Regeldifferenzen e^2. Im Gegensatz zu Optimierungsverfahren bedeutet dies aber nicht die Minimierung eines Gütekriteriums

$$J = \sqrt{\frac{1}{N} \cdot \sum_{k=1}^{N} e^2(k + n_t)}. \tag{5.24}$$

[6] Im Gegensatz dazu wäre bei Fortsetzung der Parameteridentifkation aufgrund einer neuen Trajelktorie keine Initialisierung erforderlich.

Dabei beschreibt $e(k)$ die Regeldifferenz des Abtastzeitpunktes k einer Trajektorie mit N Abtastschritten. n_t bedeutet eine totzeitbedingte Verschiebung des ausgewerteten zeitlichen Bereichs.

Die Minimierung eines Gütekriteriums J, die der *Lösung* des Gleichungssystems (5.23) entspricht, weist divergentes Verhalten auf. Auch im Fall einer konvergierenden Optimierung ergibt sich mit einem Gütekriterium nach Gleichung (5.24) eine „unruhige“ Stellfolge, da die gemessenen Regeldifferenzen - ganz im Gegensatz zum Kalman Filter - als ungestört angenommen werden.

Stattdessen wird bei Optimierungen ein Gütekriterium

$$J = \int \mathbf{x}^T \cdot \mathbf{Q} \cdot \mathbf{x} + \mathbf{u}^T \cdot \mathbf{R} \cdot \mathbf{u} dt \tag{5.25}$$

vorgegeben. Dabei beschreibt $\mathbf{x}$ den Zustandsvektor unter der Annahme, dass $\mathbf{x} = \mathbf{0}$ der Soll-Zustand ist. In zeitdiskreter Darstellung mit skalarem $\Delta u(k)$ anstelle eines Stellvektors $\mathbf{u}$ und ausschließlicher Gewichtung einer einzigen Zustandsvariablen, nämlich der Regeldifferenz e, anstelle des gesamten Zustandsvektors $\mathbf{x}$ ergibt dies

$$J = \sqrt{\frac{1}{N} \cdot \sum_{k=1}^{N} s_e^2 \cdot e^2(k + n_t) + s_u^2 \cdot u^2(k-1)} \tag{5.26}$$

Die Elemente von $\mathbf{Q}$ und $\mathbf{R}$ sind dabei mit s_e^2 und s_u^2 bezeichnet, da q und r in dieser Arbeit eine andere Bedeutung (Zeitvarianz bzw. Reglerparameter) haben.

Die Verwendung des Kalman Filters zur Schätzung der Stellgrößenfolge ist äquivalent zu einer Optimierung von Gleichung (5.23) unter Verwendung des Gütekriteriums nach Gleichung (5.26). Dabei sind die Gewichtungskoeffizienten s_e und s_u implizit durch p_{0i} und σ^2 vorgegeben. Sie können mit der Zeit variieren, wie dies ja auch für σ^2 möglich ist.

Eine Minimierung eines Gütekriteriums nach Gleichung (5.26) mit expliziter Vorgabe der Gewichtungen s_e und s_u kann auch unter Verwendung eines Kalman Filters erreicht werden. Zur Erläuterung soll zunächst noch einmal Gleichung (5.23) betrachtet werden. Dabei wird die Korrektur der Stellgrößenfolge Δu so bestimmt, dass in jedem Schritt die Systemgleichung

$$\sum_{i=1}^{n_g} g_i \cdot \Delta u(k-i) = e(k+n_t) \tag{5.27}$$

erfüllt wird.

Der Einfluss der korrigierten Stellgrößenfolge $u_{neu}(k)$ kann nun als zweite Systemgleichung dargestellt werden, ähnlich einer Regelstrecke mit zwei Ausgängen und einem Eingang. Ziel ist es, neben den Regeldifferenzen auch Abweichungen der korrigierten Stellgrößen vom Minimalwert 0 zu gewichten. In ähnlicher Weise wäre auch eine Gewichtung der Änderungen der Stellgrößen $(u_{neu}(k) - u_{neu}(k-1))$ möglich, was zu einer glatten Stellfolge führt.

Die zweite Systemgleichung soll $u_{neu}(k) = 0$ erfüllen und lautet somit

$$1 \cdot (u_{neu}(k) - u(k)) = -u(k). \tag{5.28}$$

bzw.

$$1 \cdot \Delta u(k) = -u(k). \tag{5.29}$$

Die Berücksichtigung der Gewichtungen s_e und s_u erfolgt beim inversen Kalman Filter durch Multiplikation der entsprechenden Messgleichungen.

Damit ergibt sich dann anstelle von Gleichung (5.23):

$$\begin{bmatrix} s_e \cdot g_1 & 0 & 0 & 0 \\ s_u & 0 & 0 & 0 \\ s_e \cdot g_2 & s_e \cdot g_1 & 0 & 0 \\ 0 & s_u & 0 & 0 \\ \cdots & \cdots & \cdots & 0 \\ \cdots & \cdots & s_e \cdot g_2 & s_e \cdot g_1 \\ 0 & 0 & 0 & s_u \end{bmatrix} \cdot \begin{bmatrix} \Delta u(0) \\ \Delta u(1) \\ \cdots \\ \Delta u(N-1) \end{bmatrix} = \begin{bmatrix} s_e \cdot e(n_t+1) \\ -s_u \cdot u(0) \\ s_e \cdot e(n_t+2) \\ -s_u \cdot u(1) \\ \cdots \\ s_e \cdot e(n_t+N) \\ -s_u \cdot u(N-1) \end{bmatrix} \tag{5.30}$$

Durch die Annahme von Messrauschen erfolgt indirekt auch schon eine Gewichtung der Stellgrößen. Dementsprechend wird das Gütekriterium nach Gleichung (5.26) nur grob reduziert, wenn man die tatsächliche Varianz des Messrauschens

vorgibt. Durch Eingabe eines kleinen Wertes[7] für die angenommene Varianz erreicht man dagegen die gewünschte Minimierung.

Die Gewichtung der Stellgrößen ist insbesondere dann von Bedeutung, wenn durch starke Beschleunigungen im Verlauf der Soll-Trajektorie sonst die Stellgrößenbegrenzungen erreicht werden würden. Dies vermeidet man normalerweise, indem man bei festem s_e heuristisch den Wert für s_u erhöht. Stattdessen könnte man auch direkt σ^2 erhöhen und dann den vereinfachten Ansatz nach Gleichung (5.23) anwenden. Dabei gibt es aber einen wesentlichen Unterschied: Die Kalman Filterung nach Gleichung (5.23) geht von gestörten Messwerten aus. Bei iterativer Wiederholung der Optimierung wird schließlich eine Stellfolge ermittelt, die abgesehen von stochastischen Einflüssen die Soll-Bewegung genau erreicht. Hochdynamische Soll-Funktionen führen dabei zu großen und stark schwankenden Stellsignalen. Dagegen wird bei expliziter Vorgabe der Gewichtungen und Verwendung des erweiterten Ansatzes nach Gleichung (5.30) eine Stellfolge berechnet, die große Stellsignale vermeidet, auch wenn dadurch die Soll-Bewegung nicht direkt ausgeführt werden kann. Ein derartiges Verhalten der Schätzung kann mit dem einfachen Ansatz nur erreicht werden, wenn man σ^2 erheblich erhöht oder gleich auf iterative Verbesserungen verzichtet.

Das Konvergenzverhalten des Prozesses zur Bestimmung der optimalen Bewegungsanweisungen wird am sinnvollsten im Frequenzbereich untersucht. Sowohl die Schätzung nach Gleichung (5.23) als auch die nach Gleichung (5.30) bestimmt $u_{neu}(k)$ durch Addition von Änderungen auf die vergangenen Werte $u(k)$. Voraussetzung zur Konvergenz ist damit nach Craig [56]

$$|1 - \hat{G}^{-1}(i\omega) \cdot G(i\omega)| < 1 \quad \forall\omega. \tag{5.31}$$

G und $\hat{G}$ sind dabei die tatsächliche und die geschätzte Übertragungsfunktion der Regelstrecke.

Gleichung (5.31) ist für niedrige Frequenzen immer erfüllt, da dort das Modell $\hat{G}$ mit der Strecke G annähernd übereinstimmt. Bei höheren Frequenzen ist es dagegen möglich, dass die Phase des Modells gegenüber der Strecke gedreht ist, so dass Gleichung (5.31) nicht erfüllt wird. Dies ist bei de Luca et al. [63] anhand von Ortskurven dargestellt.

Als Alternative wird in der Literatur eine Filterung vorgeschlagen, bei der durch

[7] Null ist aus Stabilitätsgründen nicht zulässig

Abschwächung der hohen Frequenzen der Stellsignale Konvergenz erreicht wird. Ein ähnliches Verfahren wird hier auch vorgeschlagen, indem explizit die Beschleunigungen durch

$$\begin{aligned} u^*(k) = \frac{u(k)}{1+2\cdot\alpha+2\cdot\beta} &+ \alpha\cdot\frac{u(k+1)-2\cdot u(k)+u(k-1)}{1+2\cdot\alpha+2\cdot\beta} \\ &+ \beta\cdot\frac{u(k+2)-2\cdot u(k)+u(k-2)}{1+2\cdot\alpha+2\cdot\beta} \end{aligned} \tag{5.32}$$

reduziert werden. Dabei sind $0 \leq \alpha, \beta \leq 1$ vorgebbare Gewichtungsfaktoren und $u^*(k)$ ist der Zwischenwert, der in Gleichung (5.23) mit $u(k)$ bezeichnet wurde. Bei diesem Ansatz wird angenommen, dass die Konvergenzprobleme in Gleichung (5.31) nur bei hohen Frequenzen ω auftreten, etwa in der Größenordnung der Abtastfrequenz ($\omega \approx 2\pi/T_0$). Die erste auftretende Schwingung, die durch das Abtastsystem erzeugt werden kann, wird durch $\alpha > 0$ gedämpft, Schwingungen mit der halben Frequenz durch $\beta > 0$. Bei $\alpha = \beta = 0$ wird keine Filterung vorgenommen. In dem Fall darf das Modell keine Phasendrehung erzeugen.

Zu der Filterung ist festzustellen, dass sie prinzipiell die Güte der im inversen Kalman Filter geschätzten Stellgrößen beeinflusst, da Stellgrößen und Regeldifferenzen nicht mehr zueinander korrelieren. Bei idealem Modell des geregelten Roboters ($\hat{G} = G$) erreicht man die optimalen Bewegungsanweisungen also mit $\alpha = \beta = 0$. Auch bei einem realem Modell sollte man die beiden Gewichtungsfaktoren sehr klein wählen. Dazu kann man bei heuristischer Vorgabe von $\beta = \alpha/2$ entweder α manuell derart einstellen, dass das System konvergiert oder eine Adaptionsregel formulieren, die α in Abhängigkeit der Modellunsicherheit anpasst.

Zur Bestimmung der optimalen Stellfolge empfiehlt es sich, *nicht* dieselbe Trajektorie zu verwenden, die schon zur Identifikation der Modellparameter ausgewertet wurde, da bei dieser Trajektorie die eigentliche Soll-Bewegung mit einem stochastischen Anregungssignal überlagert wurde. Würde man diese Trajektorie verwenden, so müsste zuerst die Anregung nach Gleichung (5.32) weggefiltert bzw. nach Gleichung (5.23) weggeschätzt werden. Stattdessen wird im Fall einer Bahnsteuerung nach Abb. 4.8 die gewünschte Trajektorie der unterlagerten Regelung ohne zusätzliche Anregung als Sollwert vorgegeben. Bei den weiteren Iterationen der Optimierung der Stellfolge wird die jeweils zuletzt ermittelte

Stellfolge selbst vorgegeben, um beim nächsten Abfahren der Bahn die Soll-Trajektorie noch genauer zu erreichen.

Bei den anderen Betriebsarten nach den Abbildungen 4.9 oder 4.10 gibt es keine Vorschriften für eine bestimmte Trajektorie. Empfehlenswert sind Trajektorien, die den erlaubten Beschleunigungs- und Geschwindigkeitsbereich des Roboters ausnutzen. Auch werden der unterlagerten Regelung des Roboters im ersten Durchlauf die Abtastwerte der Soll-Bahn vorgegeben, in weiteren Iterationen aber die durch die Vorsteuerung bzw. Regelung generierten Bewegungsanweisungen.

5.2.2 Auswertung der Gewichtsfunktion bei Messwertrückführung

Oben wurde bereits erläutert, dass die Bestimmung der optimalen Stellfolge sich von dem in Abschnitt 5.2.1 beschriebenen Ansatz unterscheidet, wenn im Regler der aktuelle Roboterzustand erfasst werden soll, um stochastische Störungen des Systems auszuregeln.

Im Fall der Adaption bei dem Ansatz mit der Rückführung von Messdaten, insbesondere bei Erfassung der aktuellen Abtastwerte der Gelenkwinkel, kann man bei einer offline Optimierung ausgehend von jedem Abtastschritt eine Teiltrajektorie definieren, die von dem gemessenen Zustand optimal zu der gewünschten Soll-Trajektorie führt. Diese Optimierung geschieht ähnlich wie in Abschnitt 5.2.1 durch Schätzung der Stellgrößendifferenzen gemäß

$$\begin{bmatrix} \hat{g}_1 & & \\ \cdots & \cdots & \\ \hat{g}_{n_g} & \hat{g}_1 & \\ & \cdots & \cdots \end{bmatrix} \cdot \begin{bmatrix} \Delta u(k) \\ \Delta u(k+1) \\ \Delta u(k+2) \\ \cdots \end{bmatrix} = \begin{bmatrix} e(k+n_t+1) \\ e(k+n_t+2) \\ e(k+n_t+3) \\ \cdots \end{bmatrix}. \tag{5.33}$$

Nach Abfahren der Trajektorie wird also zunächst die Teiltrajektorie nach Gleichung (5.33) mit $k = 0$ optimiert. Aufgrund der ermittelten Stellgrößenfolge werden die Reglerparameter in der dritten Stufe der Adaption angepasst. Dann erfolgt eine neue Stellfolgenbestimmung aufgrund der Teiltrajektorie nach Gleichung (5.33) mit $k = 1$. Auch die nun bestimmte optimale Stellfolge wird zur Adaption der Reglerparameter verwendet. In gleicher Weise wird fortgefahren, bis k das Ende der Trajektorie erreicht.

Diese Teiltrajektorien werden untereinander gleich sein, sobald die Soll-Trajektorie erreicht ist, sodass eine Begrenzung des Schätzaufwandes möglich ist. Da die Bewegungsanweisungen der Abtastschritte zur Einhaltung der Soll-Bahn am Ende der Trajektorie von den Bewegungsanweisungen zum Ausregeln der Regeldifferenz aus Schritt k unabhängig sind, reicht es, jeweils einen begrenzten zeitlichen Horizont zu betrachten. Anstelle der Optimierung der $\Delta u(k+i)$ bis zum Ende der Trajektorie bei $k = N-1$ werden jeweils nur n_o Stellgrößen optimiert, wobei diese Zahl so gewählt werden muss, dass die Begrenzung der Zahl an Gleichungen keinen merkbaren Einfluss auf die Schätzung der ersten Stellgrößendifferenz $\Delta u(k)$ hat.

$$\begin{bmatrix} \hat{g}_1 & & & & \\ \cdots & \cdots & & & \\ \hat{g}_{n_g} & & \hat{g}_1 & & \\ & \cdots & & \cdots & \\ & & \hat{g}_{n_g} & \cdots & \hat{g}_1 \end{bmatrix} \cdot \begin{bmatrix} \Delta u(k) \\ \cdots \\ \cdots \\ \cdots \\ \Delta u(k+n_o-1) \end{bmatrix} = \begin{bmatrix} e(k+n_t+1) \\ \cdots \\ \cdots \\ \cdots \\ e(k+n_t+n_o) \end{bmatrix} \tag{5.34}$$

Ein Einfluss des Zahlenwertes von n_o auf die weiteren Elemente der Stellfolge $\Delta u(k+i)$ mit $1 \leq i \leq n_o - 1$ ist allerdings nicht auszuschließen. Daher erfolgt das Training der Reglerparameter in der dritten Stufe der Adaption nur aufgrund des jeweils ersten Schrittes der k-ten Teiltrajektorie. Experimente, bei denen jeweils mehrere Abtastwerte der optimierten Stellfolge trainiert wurden, brachten keinen Vorteil, da die Soll-Trajektorie im Gleichungssystem (5.34) relativ schnell erreicht wird und ein Training weiterer Schritte ohne Regeldifferenz nur zur Adaption der Parameter der Vorsteuerung beitragen kann, nicht aber zur Bestimmung der Parameter der Messwertrückführung.

Anstelle des Gleichungssystems (5.34) wird aus Konvergenzgründen ein Ansatz verwendet, bei dem alle Gleichungen berücksichtigt werden, die von den n_o zu optimierenden Stellgrößen beeinflusst werden, also

$$\begin{bmatrix} \hat{g}_1 & & & & \\ \cdots & \cdots & & & \\ \hat{g}_{n_g} & & \hat{g}_1 & & \\ & \cdots & & \cdots & \\ & & \hat{g}_{n_g} & \cdots & \hat{g}_1 \\ & & & \cdots & \\ & & & & \hat{g}_{n_g} \end{bmatrix} \cdot \begin{bmatrix} \Delta u(k) \\ \cdots \\ \cdots \\ \cdots \\ \Delta u(k+n_o-1) \end{bmatrix} = \begin{bmatrix} e(k+n_t+1) \\ \cdots \\ \cdots \\ \cdots \\ e(k+n_t+n_o) \\ \cdots \\ e(k+n_t+n_o \\ \quad +n_g-1) \end{bmatrix}. \quad (5.35)$$

Gleichung (5.35) enthält n_o+n_g-1 Gleichungen für n_o Unbekannte, beschreibt also ein überbestimmtes System. Bei Gleichung (5.34) kann dagegen, besonders bei kleinem n_o, ein Aufklingen der Stellgrößen erfolgen, sodass $e(k+n_t+1)$ durch $\Delta u(k)$ minimiert wird, $e(k+n_t+2)$ durch $\Delta u(k+1)$ usw..

Dagegen hilft die Wahl von Gleichung (5.35) anstelle von Gleichung (5.34) nicht gegen ein zu klein gewähltes n_o. Im Extremfall $n_o = 1$ würde die Stellgröße $u_{neu}(k)$ so bestimmt, dass auch die Regeldifferenz $e(k+n_t+n_g)$ kompensiert wird, obwohl das z. B. durch $u_{neu}(k+n_g-2)$ sicherlich besser geht, weil dabei die $e(k+n_t+1), \cdots, e(k+n_t+n_g-2)$ nicht verschlechtert werden. Daraus ergibt sich die Forderung $n_o \geq n_g$. Durch Experimente lässt sich zeigen, dass $n_o = 2 \cdot n_g$ ein konservativer Ansatz ist, bei dem der Unterschied der ermittelten Bewegungsanweisungen $u_{neu}(k)$ zwischen der vollständigen Berechnung nach Gleichung (5.33) und dem vereinfachten Ansatz nach Gleichung (5.35) vernachlässigbar ist.

Der Aufwand für diese Schätzungen ist, wie bei der Steuerung oder Vorsteuerung, lediglich linear in der Zahl der Abtastschritte N. Das liegt daran, dass die ungefähr N Schätzungen einer Teiltrajektorie jeweils nur n_o Schritte optimieren, wobei $n_o \ll N$.[8]

Teiltrajektorien mit $k+n_o+n_g-1 > N$ dürfen nicht optimiert werden, da Gleichung (5.35) dabei Abtastwerte außerhalb des Definitionsbereich der Trajektorien erfordert. Das Weglassen der entsprechenden Zeilen im Gleichungssystem (5.35) ist nicht zulässig, da es beim Training der Reglerparameter in der dritten Adaptionsstufe zu Reglerparametern führt, die mit dem restlichen Training

[8]Der Aufwand ist zwar linear in N, er ist aber außerdem quadratisch in n_o. Bei Steuerungen oder Vorsteuerungen gibt es nur eine Schätzung, bei der alle N Schritte optimiert werden. Trotzdem ist der Aufwand auch dort nicht quadratisch in N, da die Matrix nur spärlich besetzt ist.

nicht konsistent sind. So entspricht z. B. die Optimierung der Teiltrajektorie mit $k = N - 1$ der oben bereits verworfenen 1-Schritt-Prädiktion.

5.2.3 Stabilitäts- und Konvergenzkriterien

Da die Schätzung der optimalen Stellgrößen die Eigenschaften des geschlossenen Regelkreises bestimmt, muss sie derart erfolgen, dass sowohl Stabilität als auch Konvergenz gewährleistet sind.

Stabilität betrifft den Roboter als Regelstrecke. Der Roboter mit unterlagerter Kaskadenregelung ist als offener Regelkreis stabil. Eine Bahnsteuerung oder Vorsteuerung beschreibt die Anregung der Strecke, ändert aber nichts am Eigenverhalten, also an der Stabilität.

Bei Rückführung von Zustandsgrößen muss die Stabilität dagegen untersucht werden. Dabei wird angenommen, dass die Strecke mit dem Modell nach Gleichung (5.16) übereinstimmt. Zur Betrachtung der Stabilität ist die Umrechnung der Parameter in die Darstellung aus Gleichung (5.15) erforderlich. Dies ist durch Berechnung einiger Abtastschritte nach einem Impuls für $u(0)$ möglich. Dadurch ergibt sich

$$\begin{aligned} \hat{g}_1 &= b_1 \\ \hat{g}_2 &= b_2 + a_1 \cdot \hat{g}_1 \\ \hat{g}_3 &= b_3 + a_2 \cdot \hat{g}_1 + a_1 \cdot \hat{g}_2 \\ &\vdots \\ \hat{g}_k &= b_k + \sum_{i=1}^{k-1} a_{k-i} \cdot \hat{g}_i. \end{aligned} \tag{5.36}$$

Meist wird man dabei nur wenige Parameter a_i und b_i ansetzen, z. B. je 2 Parameter zur Darstellung eines Systems zweiter Ordnung.

Die z-Transformierte[9] des durch die $\hat{g}_i$ beschriebenen Robotermodells lautet dann

[9] Informationen zur z-Transformation und den damit beschriebenen Stabilitätskriterien stehen z. B. bei Isermann [108].

$$\hat{G}(z) = \frac{b_1 + b_2 \cdot z^{-1} + \cdots}{1 - a_1 - a_2 \cdot z^{-1} - \cdots} \tag{5.37}$$

Bei einer z-Transformierten $R(z)$ des Reglers ist der geschlossene Regelkreis stabil, wenn die Pole von

$$\frac{R(z) \cdot \hat{G}(z)}{1 + R(z) \cdot \hat{G}(z)} \tag{5.38}$$

innerhalb des Einheitskreises liegen.

Dabei ist die Stabilität außer von den Rückführverstärkungen auch von der Abtastung abhängig, da die zeitdiskrete Streckendarstellung die Abtastzeit implizit enthält. Eine langsame Abtastung und Rückführung des Ausgangs einer stabilen Strecke ist aber nicht kritisch.

Durch eine niederfrequente Abtastung gehen aber ggf. vorhandene hochfrequente Anteile der Strecke verloren, d. h. sie werden nicht beobachtet. So kann bei einer Abtastrate von etwa 80 Hz keine Schwingung von 1000 Hz gemessen und damit auch nicht ausgeregelt werden. Als Richtwert wählt man die Abtastrate 5mal so hoch wie die höchste vorkommende Eigenfrequenz der Strecke.

Dies ist bei der in Kapitel 3 beobachteten dominanten Zeitkonstante kein Problem. Schwingungen der Roboterarmelemente erreichen dagegen unter Umständen Frequenzen, bei denen die Regelung mit der vorgegebenen Abtastrate nicht mehr möglich ist.

Neben der Stabilität des Regelkreises muss auch die Konvergenz betrachtet werden, also die Stabilität der Adaption. Dies gilt auch ohne Messwertrückführungen.

Die z-Transformierte der Adaption ergibt sich aus der z-Transformierten $G(z)$ der Strecke und der z-Transformierten der Adaptionsregel, wobei letztere, abgesehen von Filterungen, durch die inverse Modelldarstellung $\hat{G}^{-1}(z)$ gegeben ist. Die Konvergenz ergibt sich damit aus

$$\frac{\hat{G}^{-1}(z) \cdot G(z)}{1 + \hat{G}^{-1}(z) \cdot G(z)}. \tag{5.39}$$

Für $\hat{G} = G$ hat diese Gleichung keine Pole, Konvergenz ist also immer gewährleistet. Wenn Modell und reale Strecke dagegen verschieden sind, kann die Konvergenz der Adaption gefährdet sein. Dies führt zu Gleichung (5.31) und damit zur Filterung nach Gleichung (5.32).

5.3 Wahl der Reglerstruktur

In Abschnitt 5.2 wurde gezeigt, wie die Stellgrößen einer Trajektorie im Nachhinein (a posteriori) aufgrund der Regelungsfehler korrigiert werden. Ausgang dieser sogenannten Optimierung ist eine Folge von Abtastwerten der für eine gewünschte Trajektorie optimalen Stellgrößen. Aufgabe des Reglers ist es nun, die Stellgrößen im Voraus (a priori), d. h. ohne Kenntnis der später auftretenden Regelungsfehler allein aufgrund der Kenntnis der gewünschten Trajektorie und des aktuellen Roboterzustandes zu generieren.

Dabei werden drei Betriebsarten unterschieden (vergl. Abschnitt 4.3.3): Bei Bahnsteuerung einer zum Zeitpunkt der Adaption bekannten Trajektorie kann der Regler im ungestörten Fall entfallen, da dann die optimierte Stellfolge direkt ausgeführt werden kann. Bei Vorsteuerung einer beliebigen Trajektorie reicht im ungestörten Fall die alleinige Verarbeitung der Soll-Trajektorie. Dagegen muss in allen Fällen mit stochastischen Störungen auch noch der aktuelle Roboterzustand im Regler verarbeitet werden. Dies ist der Fall, wenn die Reproduzierbarkeit der unterlagerten Roboterregelung die erforderliche Genauigkeit nicht aufweist.

Die Parameter des Reglers werden durch eine neben der Identifikation und der Optimierung der Stellfolge dritte Stufe der Adaption bestimmt. Dabei werden die Parameter derart adaptiert, dass die durch die Optimierung ermittelte Stellfolge realisiert wird. Zu diesem Zweck wird nach jeder Optimierung einer Teil- oder Gesamttrajektorie die ermittelte Stellfolge auf die Reglerparameter abgebildet. Ziel ist es, Reglerparameter zu finden, die auch im Fall untrainierter Trajektorien und / oder Roboterzustände die Stellgrößen bestimmen, die die Optimierung a posteriori bestimmen würde.

Aufgrund des in Abschnitt 4.3.1 angenommenen annähernd linearen Verhaltens kann auch der Regler in erster Näherung linear angesetzt werden. Dies steht im Gegensatz zur Literatur, in der meist generalisierende Wissensbasen

(z. B. CMAC [7], MIAS [175], NPPL [122], ME9 [148], LOLIMOT [186], RBF-Netzwerke [187] oder mehrschichtige Perzeptrons) oder schaltende Modelle / Regler [48] angewandt werden. Das vorliegende Verfahren ist hybrid angelegt, d. h. grundsätzlich wird eine feste (z. B. lineare) Struktur des Reglers vorgegeben und die dadurch nicht erfassbaren Anteile werden durch gezielt eingesetzte nichtlineare Elemente repräsentiert.

Auf diese Weise werden der globale Geltungsbereich der Daten eines linearen Ansatzes mit der detaillierten Repräsentationsfähigkeit nichtlinearer Speicher kombiniert. Bei sukzessivem Wissenserwerb erlaubt zuerst die Verallgemeinerung bei spärlicher Information ein schnelles Einschwingen des Reglers auf das ungefähr richtige Verhalten, während die nichtlineare Darstellungsmöglichkeit nachher, wenn genauere Informationen vorliegen, zumindest dieselbe hohe Regelgüte wie bei rein nichtlinearen Verfahren erlaubt. In vielen Fällen wird aufgrund der günstigeren Skalierung aber ein noch besseres Verhalten erreichbar sein, da die Genauigkeit von RBF-Netzwerken oder mehrschichtigen Perzeptrons strukturbedingt begrenzt ist [259, 273].

Als nichtlineare Elemente der Regler für die einzelnen Achsen werden parallel zu den linearen Komponenten Neuronale Netze vorgesehen, um auf Kopplungen zwischen den einzelnen Achsen reagieren zu können. Zur nichtlinearen Darstellung ist ein Verfahren gefordert, das auch hochdimensionale Abbildungen repräsentieren kann. Abbildungen mit fester Quantisierung (z. B. ME7 [144], ME8 [145]) haben dabei wegen des exponentiellen Aufwands grundsätzlich Probleme. Abbildungen, die zwischen nichtlinear wirkenden und linearisierbaren Eingangsvariablen unterscheiden (z. B. ME9 [148]), sind zwar bei Ortsabhängigkeiten brauchbar, nicht aber zur Darstellung von Koppelkräften, die i. A. in vielen Variablen nichtlinear wirken. Verfahren, die durch nichtlineare Quantisierung (z. B. NPPL [122]) oder durch zufällige Zuordnung der Information auf den verfügbaren Speicherbereich (z. B. CMAC [7]) theoretisch einen höherdimensionalen Eingangsraum erlauben, konnten nicht getestet werden, versprechen aber auch keine gravierende Verbesserung.

Am aussichtsreichsten sind nichtlineare Abbildungen, bei denen die Speicherelemente *nicht* einzelne Punkte des Eingangsbereichs repräsentieren, zwischen denen linear interpoliert wird. So beschreiben die Parameter von Neuronalen Netzen *nicht* einzelne Funktionswerte, sondern Kurven, die an die darzustellende Funktion angepasst werden. Dies wird in Abschnitt 5.5.2 bei der Wahl der Netztypen näher ausgeführt.

Fuzzy Systeme bieten auch keine Ideallösung an. Sie erlauben die Repräsentation nichtlinearer Abbildungen, werden aber wegen der meist kleinen Zahl an Zugehörigkeitsfunktionen eher auf abschnittsweise lineare Zusammenhänge erfolgreich angewandt. Übliche Anwendungen unterteilen den Raum in drei bis sieben Abschnitte je Eingang, wobei für alle resultierenden Raumsegmente Stellgrößen bestimmt werden, zwischen denen interpoliert wird. Selbst wenn man für jedes Intervall die Parameter eines linearen Reglers definieren und dazwischen interpolieren würde, wäre die Darstellung doch stärker eingeschränkt als bei Neuronalen Netzen.

Die Bestimmung der Abbildung geschieht bei Fuzzy Systemen durch Vorgabe von Regeln. Dadurch ist die mögliche Zahl an unterschiedlichen Abschnitten begrenzt.

Neuro-Fuzzy Systeme (siehe z. B [81, 270]) sind dagegen in der Lage, die Abbildung adaptiv einzustellen. Damit sind verfeinerte Repräsentationen möglich. Andererseits sind auch solche Systeme nicht nötig, da der Vorteil der Parameterinterpretierbarkeit von Fuzzy Systemen im vorliegenden Entwurf nicht benötigt wird. Daher kann man direkt die gewünschte Abbildung ablegen, ohne Einteilung der Eingangsvariablen nach Zugehörigkeitsfunktionen, Mittelung des Funktionswertes (oder des Reglers) für jede Kombination der Zugehörigkeitsfunktionen und Interpolation (Defuzzifizierung) zur Gewinnung der aktuellen Stellgröße.

Abgesehen von einer Repräsentation der Kopplungen sind trotz arbeitspunktabhängiger Trägheitsmomente zumindest bei den untersuchten Robotern keine nichtlinearen Darstellungen nötig.

Dagegen ist es möglich, dass Kontaktkräfte einen Einfluss auf die Bewegung ausüben. Zur Kompensation von Kontaktkräften muss daher, bei Bewegungen mit Kontakt, außer der Positionsvorsteuerung auch aufgrund der vorhergesagten Kontaktkräfte vorgesteuert werden. Dabei reicht für die Kraftvorsteuerung (Kontaktkraftkompensation) i. A. ein linearer Ansatz. Kräfte stellen also eine besondere Art von Sensordaten dar, da sie neben der Beschreibung der Soll-Bahn auch direkt die Bewegung beeinflussen, also auch direkt im Regler (bzw. in der Vorsteuerung) berücksichtigt werden müssen.

Die gesamte Vorsteuerung ist in Abb. 5.4 für eine einfache Konfiguration dargestellt. Dabei ergeben sich die Bewegungsanweisungen q_{ci} der Achsen i jeweils aus den Anteilen einer linearen Positionsvorsteuerung $q_{ci,pv}$, eines Neuronalen

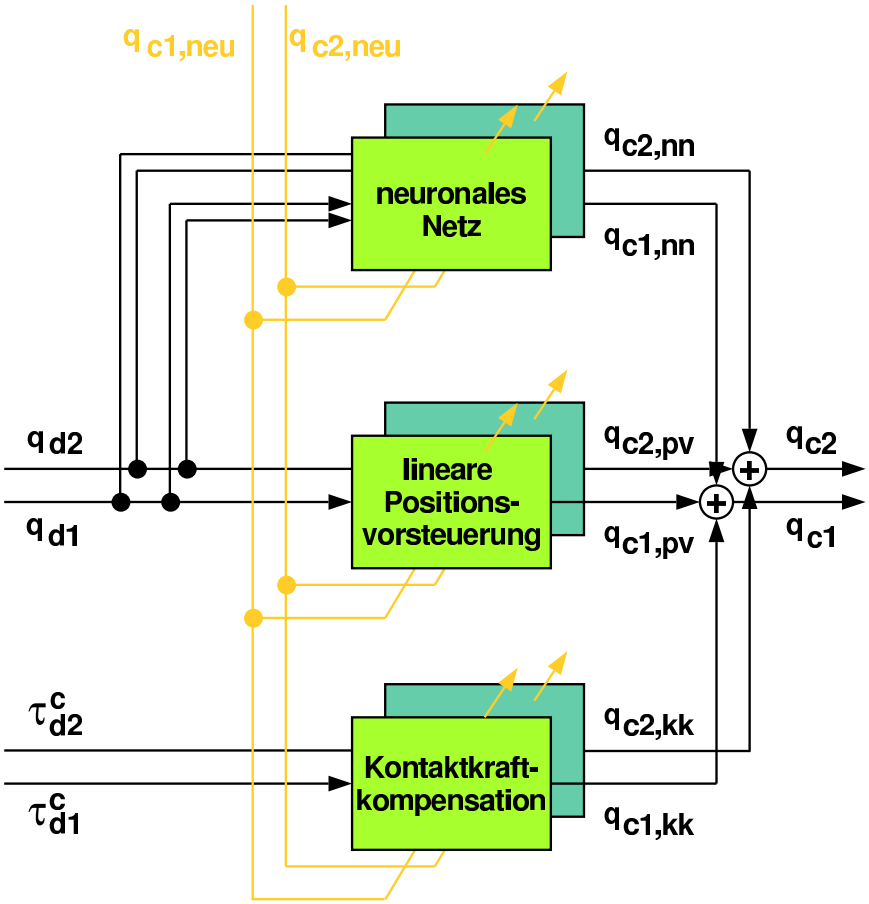


Abbildung 5.4: Struktur der Vorsteuerung (ohne Messwertrückführung) für einen Roboter mit 2 Achsen (hell gezeichnet ist die Adaption aufgrund der optimierten Stellfolge)

Netzes mit Ausgang $q_{ci,nn}$ und einer Kontaktkraftkompensation $q_{ci,kk}$. Bei Messwertrückführung müssen entsprechend noch mehr Funktionsblöcke vorgesehen werden.

Damit die Gesamtstrukturen von Abb. 4.9 und Abb. 4.10 trotz der Realisierung der Vorsteuerung nach Abb. 5.4 gültig bleiben, müssen die dortigen Variablen u, y und w nun nicht mehr als einzelne Achssignale, sondern als Vektoren interpretiert werde, die einerseits alle Achsen beinhalten und andererseits neben den Positionswerten auch die Kraftwerte. Somit ist w der Vektor der Soll-Bahnpunkte $\mathbf{q}_d$ und Soll-Kräfte[10] τ_d^c, y ist der Vektor der Ist-Bahnpunkte $\mathbf{q}$ und der Kräfte τ^c, u ist der Vektor der kommandierten Bahnpunkte $\mathbf{q}_c$ (Kräfte können bei üblichen Roboterschnittstellen nicht explizit vorgegeben) und u_{neu} wird durch optimierte Folge der Kommandos $\mathbf{q}_{c,neu}$ ersetzt. Wegen der Beschreibung der Positionen im Achsraum müssen die i. A. kartesisch messbaren Kräfte $\mathbf{F}$ durch

[10]Genau genommen beschreibt τ_d^c den Anteil der Achsmomente, der die kartesische Soll-Kontaktkraft $\mathbf{F}_d$ verursachen soll.

die transponierte Jacobi-Matrix $\mathbf{J^T}$ in den Achsraum transformiert werden.

Der Ansatz einer Vorsteuerung nach Abb. 5.4 macht trotz des linearen Modells einen Sinn, da die Adaption inkrementell erfolgt. Dadurch kann, ausgehend von einer grob adaptierten Bahnsteuerung, auch die Optimierung mit einem linearen Modell eine Korrektur ergeben, die die Bahntreue bei derselben Testbahn weiter verbessert. Ob dies auch bei Verwendung einer Vorsteuerung oder Regelung und auch bei anderen Bahnen gilt, hängt von der Struktur des Reglers ab, die deshalb unter Umständen erheblich aufwendiger sein muss als die Struktur des Modells.

Zur Darstellung der Dynamik von Systemen in Neuronalen Netzen gibt es grundsätzlich zwei Möglichkeiten [188]. Im einen Fall wird die Zeitabhängigkeit extern vorgegeben, indem Eingangsgrößen von verschiedenen Zeitpunkten in einer statischen Abbildung die Ausgangswerte liefern. Dies ist z. B. bei vorwärtsgerichteten Neuronalen Netzen der Fall. Nach der Verschaltung der Eingänge unterscheiden Nelles et al. [188] zwischen Modellen mit Ausgangsrückkopplung, ähnlich einer Übertragungsfunktion

$$\begin{aligned} \hat{y}(k) \;=\; & f(u(k), u(k-1), \cdots, u(k-n_u+1), \\ & y(k-1), y(k-2), \cdots, y(k-n_y)), \end{aligned} \tag{5.40}$$

Modellen mit endlicher Impulsantwort

$$\hat{y}(k) = f(u(k), u(k-1), \cdots, u(k-n+1)) \tag{5.41}$$

und Modellen mit orthonormalen Basisfunktionen

$$\hat{y}(k) = f(F_0 \cdot u(k), F_1 \cdot u(k), \cdots, F_n \cdot u(k)), \tag{5.42}$$

bei denen die $F_i \cdot u(k)$ jeweils fest definierte Zeitfunktionen der Eingangsgröße beschreiben [246].

Bei der anderen Möglichkeit zur Berücksichtigung der Zeitabhängigkeit dienen als Eingangsgrößen nur die aktuellen Werte, die Abbildungen enthalten aber interne Zustände. Dies ist bei zeitverzögernden [118] oder rekurrenten Neuronalen Netzen der Fall [187, 188, 267]. Vorteil der rekurrenten Abbildungen

ist, dass nicht vorgegeben werden muss, wie viele Abtastschritte jeweils betrachtet werden. Nachteil ist, dass die durch ein rekurrentes System darstellbaren Abbildungen so vielfältig sind, dass das Training sehr aufwendig ist (siehe z. B. [12, 133, 212, 255]). Daher wird die statische Abbildung mit vorgegebener Zeitabhängigkeit gewählt.

Bei Problemen mit wenigen Eingängen ist eine Darstellung nach Gleichung (5.40) oder (5.41) sinnvoll, wobei die Eingänge im nächsten Abschnitt festgelegt werden. Bei im Vergleich zur Trägheit schneller Abtastung würden sich dadurch allerdings sehr viele Eingänge ergeben, was das Approximationsproblem verschärft. In dem Fall sind orthonormale Basisfunktionen nach Gleichung (5.42) empfehlenswert, sofern die Abtastrate nicht reduziert werden kann.

Es ist günstig, die linearen und nichtlinearen Komponenten der Regler parallel anzuordnen, da dadurch eine voneinander unabhängige Adaption der Funktionsblöcke ermöglicht wird. Aus dem gleichen Grund empfiehlt sich auch eine parallele Struktur der linearen Funktionsblöcke Vorsteuerung und Messwertrückführung.

Die Adaption der Parameter geschieht bei den einzelnen Reglerelementen grundsätzlich gleich. Sie werden für jeden Ausgang i getrennt derart bestimmt, dass der quadratische Fehler zwischen dem Ausgang $q_{ci}(k)$ und dem Sollwert $q_{ci,neu}(k)$[11] minimal wird. Dabei wird zunächst ohne Kraftkontakt die lineare Positionsvorsteuerung $q_{ci,pv}(k)$ trainiert, so dass $q_{ci,pv}(k) = q_{ci,neu}(k)$. Nach Einschwingen der linearen Positionsvorsteuerung noch verbleibende Fehler werden durch Modifikation der Gewichte der Neuronalen Netze so minimiert, dass $q_{ci,nn}(k) = q_{ci,neu}(k) - q_{ci,pv}$. Zuletzt wird die Kraftvorsteuerung (Kontaktkraftkompensation) $q_{ci,kk}(k)$ durch Verwendung einer Trajektorie mit Kraftkontakt trainiert. Dabei wird der Fehler von $q_{ci,kk}(k) = q_{ci,neu}(k) - q_{ci,pv} - q_{ci,nn}$ minimiert.

Die einzelnen Parameteradaptionen erfolgen durch Anwendung eines Approximationsverfahrens, ähnlich wie bei der Identifikation der Modellparameter in Abschnitt 5.1. Bei den linearen Reglerelementen kann auch der gleiche Algorithmus verwendet werden. Die Adaption der Neuronalen Netze kann durch Backpropagation erfolgen. Günstiger sind jedoch andere Verfahren, z. B. EKFNet [151] nach Anhang B.

[11] $q_{ci,neu}(k)$ ist das i-te Element des in der Optimierungsstufe bestimmten Vektors $\mathbf{q}_{c,neu}(k)$.

5.4 Adaption der linearen Teilkomponenten des Reglers

Sofern die Aufgabe darin besteht, die Stellgrößen für eine feste Trajektorie zu bestimmen, ist das Problem durch Abschnitt 5.2 gelöst. Die optimalen Stellgrößen aus Abschnitt 5.2.1 müssen lediglich als Stellfolge abgespeichert werden (siehe auch Abb. 4.8).

Soll das Verhalten dagegen auch auf andere Soll-Trajektorien übertragbar sein, so muss ein weiterer Adaptionsschritt folgen. Dies wird zunächst für den Fall der Positionsvorsteuerung behandelt, bevor die Variante der Messwertrückführung mit der Optimierung aus Abschnitt 5.2.2 folgt. Die in Abb. 5.4 erwähnte Kraftvorsteuerung folgt in Abschnitt 5.4.3, das Neuronale Netz in Abschnitt 5.5.

In allen Fällen wird eine parametrische „Regel“ zur Modifikation von Bahnkommandos bestimmt, die online die Stellgrößen zum Abfahren einer Soll-Bahn so abändert, dass die Bahnfehler minimal werden.

5.4.1 Bestimmung der Parameter zur Vorsteuerung ungestörter Trajektorien

Im Fall einer Positionsvorsteuerung wird die zukünftige Soll-Trajektorie durch ein Vorfilter gewichtet. Dieses Vorfilter repräsentiert die Umkehrung des Modells. Das legt einen ähnlichen Ansatz wie bei der Modellgleichung (5.16) oder (5.17) nahe. Die Schnittstellen lassen sich aus Abb. 4.9 ablesen und sind in Abb. 5.5 noch einmal aufgeführt.

Allgemein kann man die Vorsteuerung als

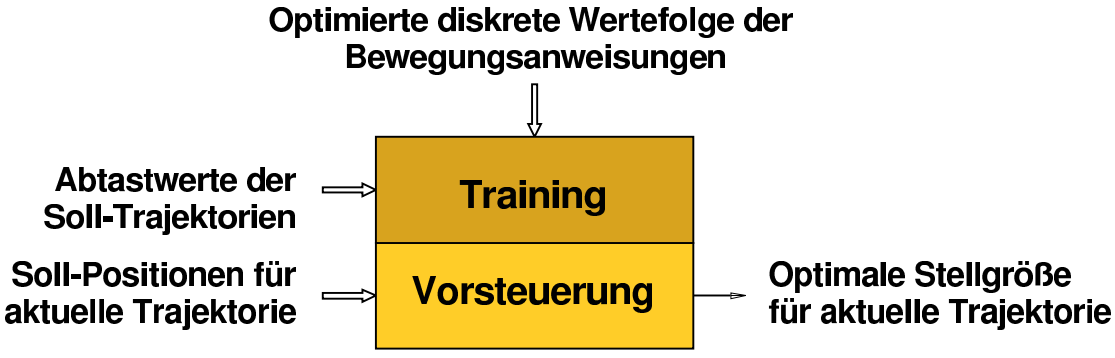


Abbildung 5.5: Schnittstellen der Adaption der Vorsteuerung

$$u(k) - u_0 = r_{w1} \cdot (w(k+1) - w_0) + r_{w2} \cdot (w(k+2) - w_0) + \cdots \tag{5.43}$$

ansetzen. Dabei sind die r_{wi} die Parameter des Vorfilters und w bezeichnet die Achs-Sollwerte an den einzelnen Abtastzeitpunkten.

Unter der Annahme einer stabilen Strecke, bei der die Gewichtsfunktion sich durch wenige Elemente nähern lässt, kann man unter Verwendung der Totzeit n_t das Vorfilter analog zu Gleichung (5.17) vereinfachen:

$$u(k) = w(k) + r_{w1} \cdot (w(k+n_t+1) - w(k)) + \cdots + r_{wn_w} \cdot (w(k+n_t+n_w) - w(k)) \tag{5.44}$$

Dabei reicht als Zahl der Parameter des Vorfilters

$$n_w = n_o = 2 \cdot n_g \tag{5.45}$$

sicher aus, da eine Stellgröße zum Zeitpunkt k keinen direkten Einfluss auf die Soll-Positionen nach dem Zeitpunkt $k + n_t + n_g$ hat. $n_w > n_g$ ist sinnvoll, um prädiktiv auf Sollwertänderungen zu reagieren, damit die folgenden Stellgrößen glatter sein können.

Bei der Anwendung des Vorfilters ergibt sich aus Gleichung (5.44) online die Stellgröße u, also die kommandierte Position, die an die Robotersteuerung weitergegeben wird (siehe Abb. 4.9). Sie besteht aus der Soll-Position $w(k)$ und der eigentlichen Vorsteuerung.

Bei der Adaption ist $u(k)$ die korrigierte Stellgröße $u_{neu}(k)$ aus Gleichung (5.23). Die Adaption des Vorfilters ist also auch eine Parameterschätzung, wie die Identifikation in Abschnitt 5.1. Dabei sind sowohl die Koeffizienten $(w(k+i) - w(k))$ als auch die erforderliche Vorsteuerung $(u(k) - w(k))$ gegeben und die Parameter r_{wi} gesucht.

Durch die Subtraktion von $w(k)$ wird erreicht, dass die Koeffizienten der Reglerparameter an allen Punkten des Arbeitsraums in der gleichen Größenordnung liegen. Dadurch ist auch hier eine ortsunabhängige linearisierte Schätzung erlaubt. Diese wird, wie bei der Identifikation, durch ein rekursives Kalman Filter (Anhang A.1) realisiert.

Es ist allerdings keine Anregung durch Addition stochastischer Werte auf die Soll-Positionen (vor der Optimierung) nötig, da es bei der Vorsteuerung nicht darauf ankommt, Parameter zu bestimmen, die weiter verwendet werden. Während die Elemente der Gewichtsfunktion in Gleichung (5.23) eingesetzt werden, dienen die r_{wi} nur zur Bestimmung von $u(k)$ nach Gleichung (5.44). Dabei bewirken korrelierte Koeffizienten $w(k+i) - w(k)$ zwar theoretisch verfälschte Reglerparameter r_{wi}, dies hat aber keinen Einfluss auf $u(k)$.

Für ein Vorfilter, das für beliebige Trajektorien anwendbar sein soll, müssen allerdings die Trainingstrajektorien ein genügend großes Spektrum an unterschiedlichen Bahncharakteristiken enthalten. Das ist bei Industrierobotern gegeben, da ihre Bahnen kartesisch definiert und in Gelenkwinkelbahnen umgerechnet werden.

Der Ablauf der Adaption einer trajektorienunabhängigen Vorsteuerung ist in Abb. 5.6 dargestellt. Dabei kann die Trajektorie jedes Mal unterschiedlich sein. In der Praxis wird das gesamte Training aber wohl an wenigen Trajektorien ausgeführt.

5.4.2 Bestimmung der Parameter zur Messwertrückführung für gestörte Trajektorien

Im Fall der Messwertrückführung sieht die Adaption des Reglers ähnlich aus. Dies wird nun erläutert:

Man muss zunächst definieren, welche Eingangsgrößen das System hat. Da es sich um eine vollständige Zustandsregelung handeln soll, müssen der Ist-Zustand und der Soll-Zustand verglichen werden. Die Schnittstellen lassen sich auch aus Abb. 4.10 ablesen und sind in Abb. 5.7 noch einmal aufgeführt. Dabei wird von einer anderen Streckendarstellung, der diskreten Übertragungsfunktion, ausgegangen, da die Gewichtsfunktionsdarstellung nach Gleichung (5.17) die Regelabweichung nicht erfasst.

Der Zustand des Systems lässt sich durch die abgegebenen Bewegungsanweisungen und Messwerte zu verschiedenen Zeitpunkten ausdrücken. So beschreiben bei einer Modellgleichung[12]

[12]Gleichung (5.46) ist eine allgemeinere Form von Gleichung (5.2) mit $1 + n_a + n_b$ Parametern.

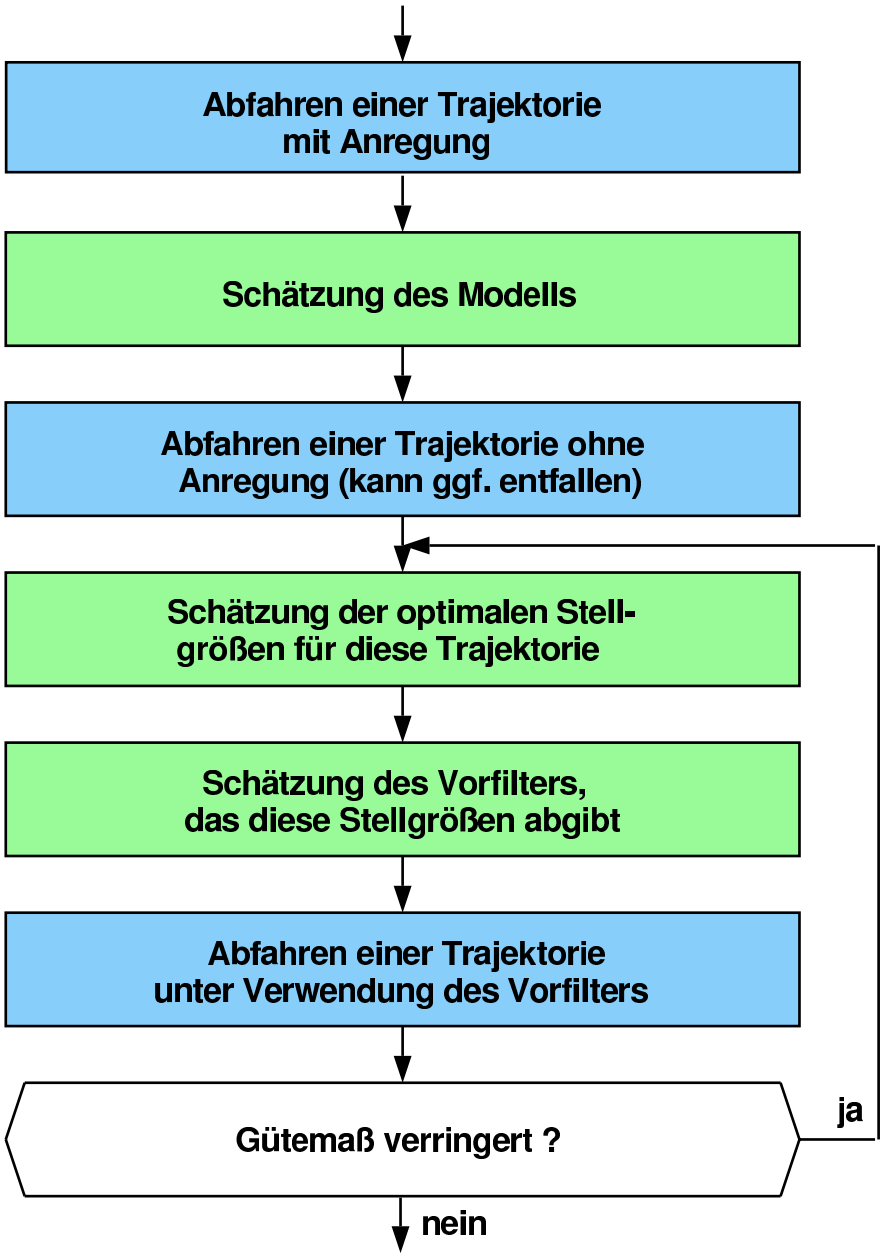


Abbildung 5.6: Ablauf der Adaption einer trajektorienunabhängigen Vorsteuerung

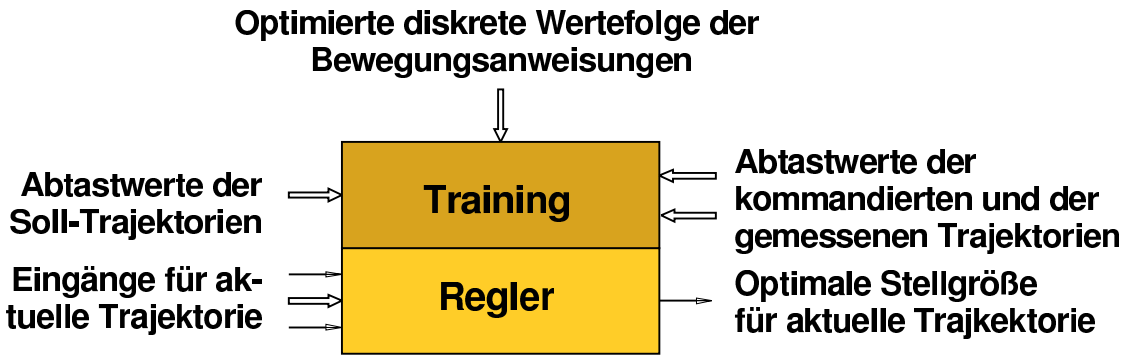


Abbildung 5.7: Schnittstellen der Regleradaption bei Messwertrückführung

$$\begin{aligned} y(k+1) &= b_0(k) + b_1 \cdot u(k-n_t) + b_2 \cdot u(k-n_t-1) + \cdots \\ &\qquad + b_{n_b} \cdot u(k-n_t-n_b+1) \\ &+ a_1 \cdot y(k) + a_2 \cdot y(k-1) + \cdots + a_{n_a} \cdot y(k-n_a+1) \end{aligned} \tag{5.46}$$

die Variablen $u(k) \cdots u(k-n_t-n_b+1)$ und $y(k) \cdots y(k-n_a+1)$ den Zustand zum Zeitpunkt $k+1$. Durch Auflösung von Gleichung (5.46) nach $y(k-n_a+1)$ kann man den Zustand zum Zeitpunkt $k+1$ auch durch $u(k) \cdots u(k-n_t-n_b+1)$ und $y(k+1) \cdots y(k-n_a+2)$ beschreiben. Ein Zustandsregler bestimmt $u(k)$ aufgrund des Zustandes zum Zeitpunkt k, der z. B. durch $u(k-1) \cdots u(k-n_t-n_b)$ und $y(k) \cdots y(k-n_a+1)$ gegeben ist.

Zusammen mit der Vorsteuerung hat der Regler danach in Anlehnung an Gleichung (5.44) folgende Form (siehe auch Abb. 5.8[13]):

$$\begin{aligned} u(k) = w(k) &+ r_{w1} \cdot (w(k+n_t+1) - w(k)) + \cdots \\ &\qquad + r_{wn_w} \cdot (w(k+n_t+n_w) - w(k)) \\ &+ r_{e1} \cdot (y(k) - w(k)) + \cdots \\ &\qquad + r_{en_e} \cdot (y(k-n_e+1) - w(k)) \\ &+ r_{u1} \cdot (u(k-1) - w(k)) + \cdots \\ &\qquad + r_{un_u} \cdot (u(k-n_u) - w(k)) \end{aligned} \tag{5.47}$$

Diese Darstellung ergibt sich durch Verschiebung des Nullpunkts in den Wert

[13] Zur besseren Vergleichbarkeit mit Abb. 5.4 werden die allgemeinen Variablen w, y und u durch q_d, q und q_c ersetzt. Dabei werden der Übersichtlichkeit halber die Gelenkindizes weggelassen, da jeweils nur eine Komponente betrachtet wird.

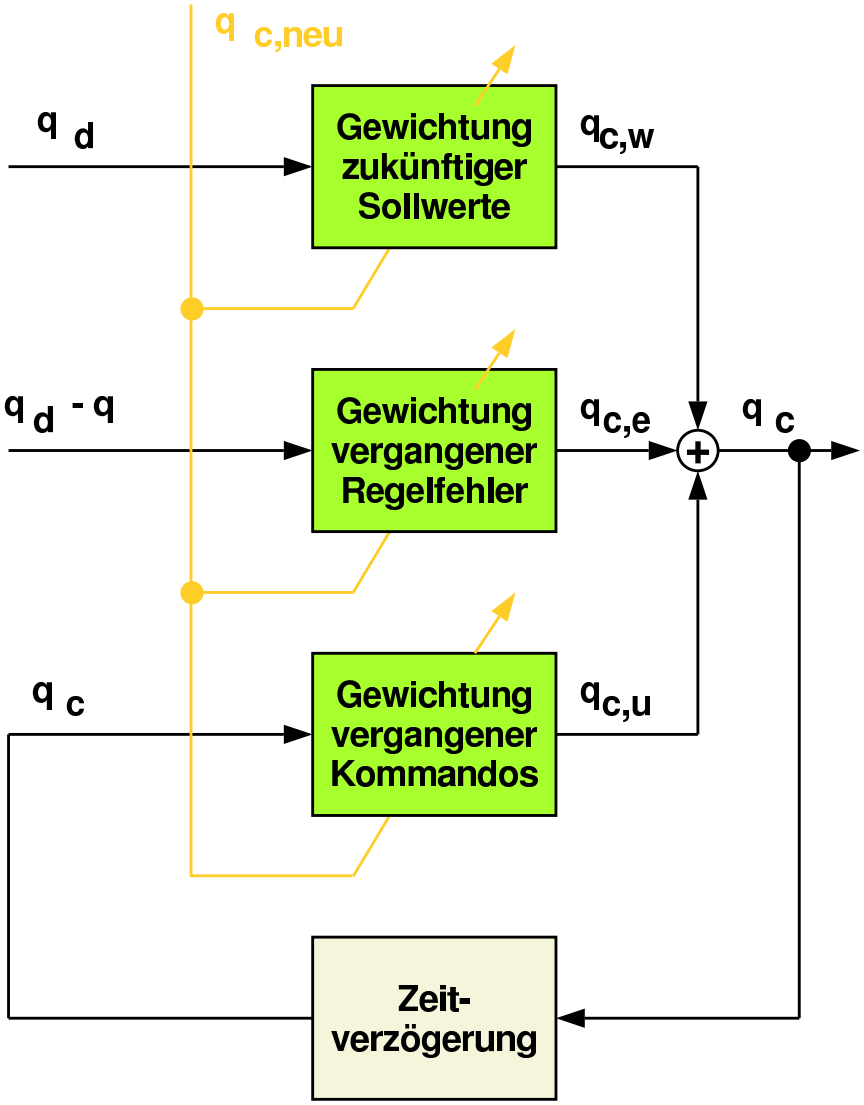


Abbildung 5.8: Struktur der zusätzlich zur internen Kaskadenregelung vorgeschlagenen linearen Positionsregelung und -vorsteuerung für eine Roboterachse (hell gezeichnet ist die Adaption aufgrund der optimierten Stellfolge; die Nullpunktverschiebung um $w(k)$ ist nicht dargestellt)

$w(k)$. Der erste Index der Reglerparameter r steht für den Typ der berücksichtigten Variable. Die r_{wi} gewichten also die zukünftigen Sollpositionen, die r_{ei} die vergangenen Regeldifferenzen und die r_{ui} die abgegebenen Bewegungsanweisungen. Der zweite Index ist schließlich eine Nummerierung, aus der sich der Abtastschritt der berücksichtigten Variable ergibt. So gewichtet beispielsweise r_{w2} bei der Berechnung von $u(k)$ den Unterschied zwischen der Variable $w(k+n_t+2)$ und dem mitgeführten Arbeitspunkt $w(k)$. Die Zahl der Reglerparameter für einen Variablentyp wird durch n mit dem Index des Variablentyps bezeichnet, z. B. n_w für die Zahl der Reglerparameter r_{wi}.

Die Zahl der zusätzlich zur Vorsteuerung verwendeten Reglerparameter be-

stimmt man durch

$$n_e = n_a \tag{5.48}$$

und

$$n_u = n_b + n_t. \tag{5.49}$$

Damit lässt sich der Ablauf des Training eines Reglers mit Messwertrückführung im Gegensatz zur Vorsteuerung durch Abb. 5.9 beschreiben.

Bezüglich der Empfindlichkeit der resultierenden Bewegung gegenüber ungünstig gewählten Parametern oder Störungen muss bemerkt werden, dass Regelungen im Vergleich zu Steuerungen von Tiefpasssystemen bei hohen Frequenzen empfindlicher sind [86]. Da die Modellinformation gerade bei hohen Frequenzen unsicher ist, kann sich deshalb ein unter Umständen instabiles Verhalten ergeben. Auf der anderen Seite wurde in der Anforderungsanalyse von Abschnitt 4.3 festgelegt, dass der Regler in erster Linie optimal, also nahe an der Stabilitätsgrenze, und erst in zweiter Linie robust sein soll. Es ist also darauf zu achten, dass die Parameter der Adaption geeignet gewählt werden.

Der Empfindlichkeit bezüglich Störungen wie unmodelierten Nichtlinearitäten oder numerischen Ungenauigkeiten kann nur durch ausgiebiges Training begegnet werden, wodurch sich Robustheit gegenüber den Störungen ergibt.

5.4.3 Einfluss von externen Kräften

Besser als jede Regelung ist sicherlich eine Störgrößenaufschaltung, also die Erfassung von Regeldifferenzen bei deren Entstehen. Die Ursachen von Bahnabweichungen sind vorhersagbar, wenn der Grund dafür im Kraftkontakt des Greifers oder Werkzeugs mit der Umwelt liegt. Dann sind auch die Zwangskräfte sensorisch erfassbar. Berührungen der Armelemente mit Hindernissen in einer Arbeitszelle sind nur schwer erfassbar.

Es sei erwähnt, dass es in diesem Abschnitt um eine durch Kräfte möglichst wenig verfälschte Bahn geht, die positionsmäßig vollständig gegeben ist (wie in

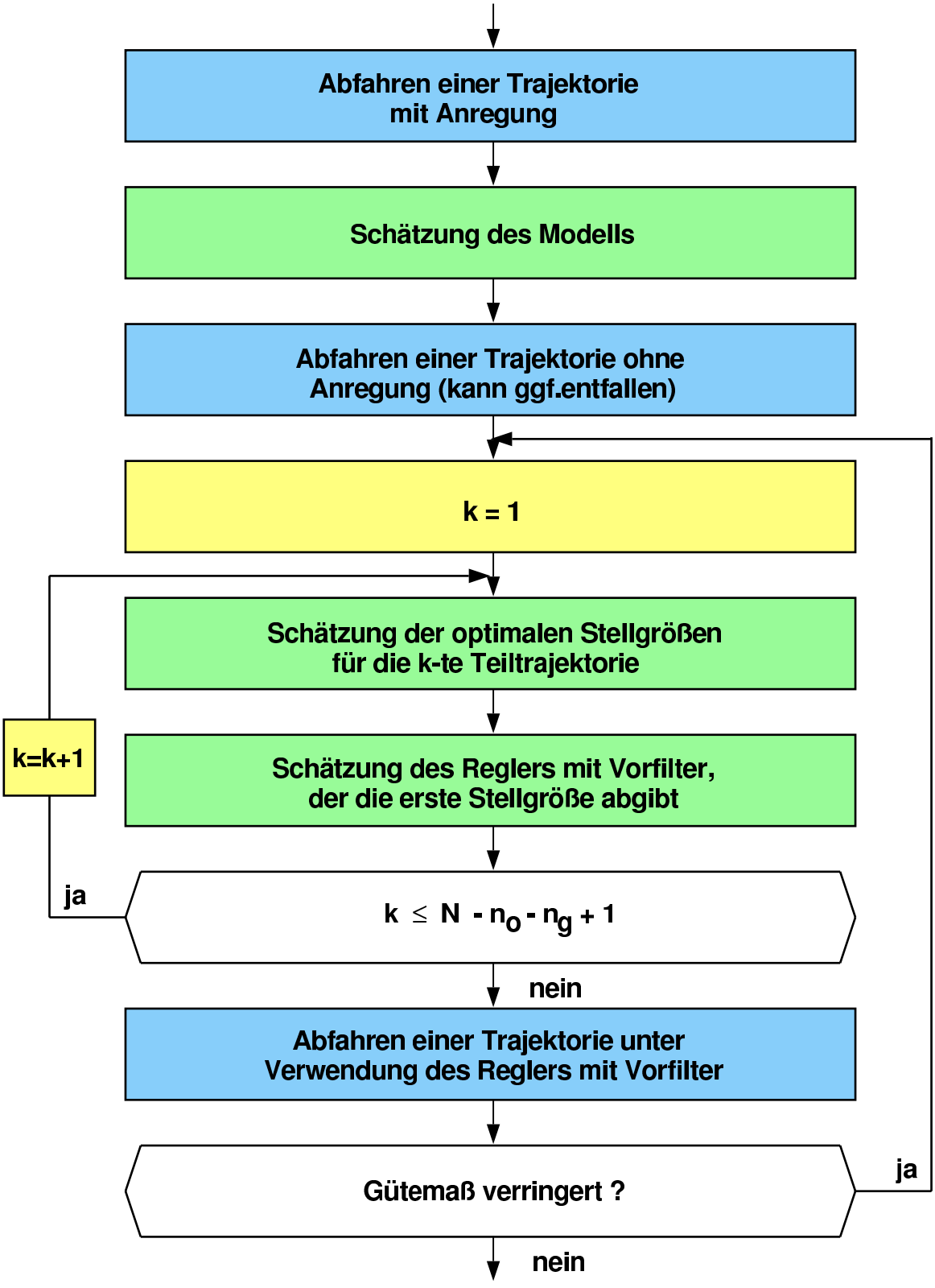


Abbildung 5.9: Ablauf der Adaption mit Messwertrückführung

[269]). Das hat nichts mit der Modifikation einer Soll-Bahn wegen auftretender Kräfte zu tun. Letzteres wurde in Abschnitt 3.3.1 betrachtet.

Solche Kontaktkräfte kommen z. B. vor, wenn eine horizontale Bahn auf einer Tischfläche erfolgt oder wenn eine kreisförmige Bewegung wie in Abschnitt 3.2.2 in einem Ring vorgeschrieben ist, der etwas kleiner als die Soll-Bahn ist.

Zur Berücksichtigung der Kontaktkräfte und -momente muss zunächst einmal der Einfluss auf die einzelnen Achsen festgestellt werden. Dies geschieht in 3 Schritten:

Im 1. Schritt werden die gemessenen Kräfte $\mathbf{F}$ und Momente $\mathbf{M}$ auf die Werkzeugspitze (tool center point) bezogen. Dies bedeutet neben einer Anpassung der Orientierung (soweit nötig) die Umrechnung der Momente auf den anderen Bezugspunkt. Bei momentenfreiem Kontakt im tool center point bleiben dabei nur Kräfte übrig.

Im 2. Schritt werden die Kräfte $\mathbf{F}_{tcp}$ und Momente $\mathbf{M}_{tcp}$ ins kartesische Inertialsystem umgerechnet. Dies geschieht durch Multiplikation mit der Orientierungsmatrix $\mathbf{R}_{tcp}$:

$$\mathbf{F}_x = \mathbf{R}_{tcp} \cdot \mathbf{F}_{tcp} \tag{5.50}$$

$$\mathbf{M}_x = \mathbf{R}_{tcp} \cdot \mathbf{M}_{tcp} \tag{5.51}$$

Im 3. Schritt kann der ins Inertialsystem umgerechnete Sensorvektor mit der Jacobi-Matrix $\mathbf{J}$ in die Momente τ^c umgerechnet werden, die aufgrund des Kontakts in den Gelenken zusätzlich auftreten:

$$\tau^c = \mathbf{J}^T \cdot \begin{bmatrix} \mathbf{F}_x \\ \mathbf{M}_x \end{bmatrix} \tag{5.52}$$

Eine optimale Störgrößenaufschaltung ist die Vorsteuerung aufgrund der zukünftigen Gelenkmomente. Im Gegensatz zur Soll-Bahn sind die Gelenkmomente vor dem Abfahren der Bahn aber nicht für alle Zeitpunte bekannt. Man muss sie also vorhersagen. Dabei reicht es nicht, das transformierte Kraftsignal zu extrapolieren, da die Gewichtung von Daten, die aus Messwerten extrapoliert

wurden, der Gewichtung der Messwerte selbst vom Informationsgehalt gleichkommt, bei Adaption der Gewichtungskoeffizienten also zum gleichen Ergebnis führt.

Kontaktkräfte sind nur vorhersagbar, wenn man die Bahn schon einmal abgefahren hat oder wenn ein genaues Umweltmodell existiert. Bei mehrmaligem Abfahren einer Bahn braucht man aber keine Vorsteuerung mit zahlreichen Parametern zu adaptieren, sondern es reicht, entsprechend Abschnitt 5.2.1 die bahnspezifische Steuerung zu bestimmen, die sowieso genauer ist. Diese Steuerung enthält auch die Reaktion auf systematische Störungen.

In den wenigen Fällen, in denen die Kraft geeignet vorhersagbar ist, z. B. bei Bearbeitungsaufgaben, bei denen die Soll-Bahn so geplant wurde, dass ein definierter Betrag der Kraft erreicht wird, kann man mit den vorhergesagten Werten eine Kraftvorsteuerung durchführen. Dabei wird angenommen, dass die aktuelle wirkende Kontaktkraft τ^c und die Soll-Kontaktkraft τ_d^c von der oberen Ebene der hierarchischen Struktur aus Abschnitt 4.2 identisch sind.

Da konstante Kräfte bei korrekt eingestellter Kaskadenregelung durch diese kompensiert werden, braucht man sie nicht vorzusteuern. Man kann stattdessen, ähnlich wie bei den Positionen, nur die Änderungen gegenüber einem mitgeführten Nullpunkt vorsteuern. Dieser Nullpunkt beschreibt den stationären Fall, in dem keine Vorsteuerung nötig ist.

Zusammen mit der Positionsvorsteuerung aus Abschnitt 5.4.1 (Gleichung (5.44)) ergibt dies folgenden Ansatz: [14]

$$\begin{aligned} q_c(k) = q_d(k) \;+\; & r_{w1} \cdot (q_d(k+n_t+1) - q_d(k)) + \cdots \\ & \qquad + r_{wn_w} \cdot (q_d(k+n_t+n_w) - q_d(k)) \\ +\; & r_{c1} \cdot (\tau_d^c(k+n_t+1) - \tau_d^c(k)) + \cdots \\ & \qquad + r_{cn_c} \cdot (\tau_d^c(k+n_t+n_c) - \tau_d^c(k)) \end{aligned} \tag{5.53}$$

Wie bei Gleichung (5.47) steht der erste Index der Reglerparameter r für den Typ der berücksichtigten Variable. Die r_{wi} gewichten also die zukünftigen Sollpositionen und die r_{ci} die zukünftigen Kontaktkräfte. Die Zahl der Reglerpa-

[14] Da die folgenden Gleichungen sich auf die speziellen Größen beim Roboter beziehen, werden die allgemeinen Variablen w, y und u durch die spezifischen Größen beim Roboter ersetzt. Dabei werden der Übersichtlichkeit halber die Gelenkindizes weggelassen, da jeweils nur eine Komponente betrachtet wird.

rameter für einen Variablentyp wird durch n mit dem Index des Variablentyps bezeichnet, z. B. n_w für die Zahl der Reglerparameter r_{wi}. Man beachte aber, dass Gleichung (5.53) nur eine Achse betrachtet, dass also für jede Achse n_w Reglerparameter r_{wi} und n_c Reglerparameter r_{ci} vorgesehen werden, ggf. sogar mit unterschiedlichen Werten für n_w und n_c.

Abgesehen von der Nullpunktverschiebung um $q_d(k)$ bzw. $\tau_d^c(k)$ ist diese Struktur in Abb. 5.4 (zusammen mit nichtlinearen Elementen der Vorsteuerung) dargestellt.

Bei der Betriebsart mit Messwertrückführung (Abschnitt 5.4.2) enthält der Regler (Gleichung(5.47)) zusätzlich noch eine Kraftrückführung. Dabei kann man aber die tatsächlichen Momente τ^c anstelle der Soll-Momente τ_d^c verwenden. Im Gegensatz zu den Positionen wirken bei den Kontaktkräften aufgrund von Sättigungserscheinigungen ggf. nicht nur die Änderungen, sondern auch die Absolutwerte selbst. Damit ergibt sich anstelle von Gleichung (5.47) (siehe auch Abb. 5.10):

$$\begin{aligned} q_c(k) = q_d(k) \;+\; & r_{w1} \cdot (q_d(k+n_t+1) - q_d(k)) + \cdots \\ & \qquad + r_{wn_w} \cdot (q_d(k+n_t+n_w) - q_d(k)) \\ +\; & r_{c1} \cdot (\tau_d^c(k+n_t+1) - \tau_d^c(k)) + \cdots \\ & \qquad + r_{cn_c} \cdot (\tau_d^c(k+n_t+n_c) - \tau_d^c(k)) \\ +\; & r_{e1} \cdot (q(k) - q_d(k)) + \cdots \\ & \qquad + r_{en_e} \cdot (q(k-n_e+1) - q_d(k)) \\ +\; & r_{u1} \cdot (q_c(k-1) - q_d(k)) + \cdots \\ & \qquad + r_{un_u} \cdot (q_c(k-n_u) - q_d(k)) \\ +\; & r_{f1} \cdot \tau^c(k) + \cdots + r_{fn_f} \cdot \tau^c(k-n_f+1) \end{aligned} \tag{5.54}$$

Die n_f Reglerparameter zur Gewichtung der vergangenen Kontaktkräfte (bzw. -momente) der jeweiligen Achse werden durch r_{fi} bezeichnet.

Man kann den Anteil der Positionsrückkopplung durch $n_e = n_u = 0$ vernachlässigen, wenn die Positionsregelung aufgrund der gut eingestellten roboterinternen Kaskadenregelung nicht nötig erscheint.

Bei einer guten Kaskadenregelung der Achspositionen erweist sich aber auch die Kraftrückführung als wenig erfolgreich. Dann reagiert nämlich die unterlagerte

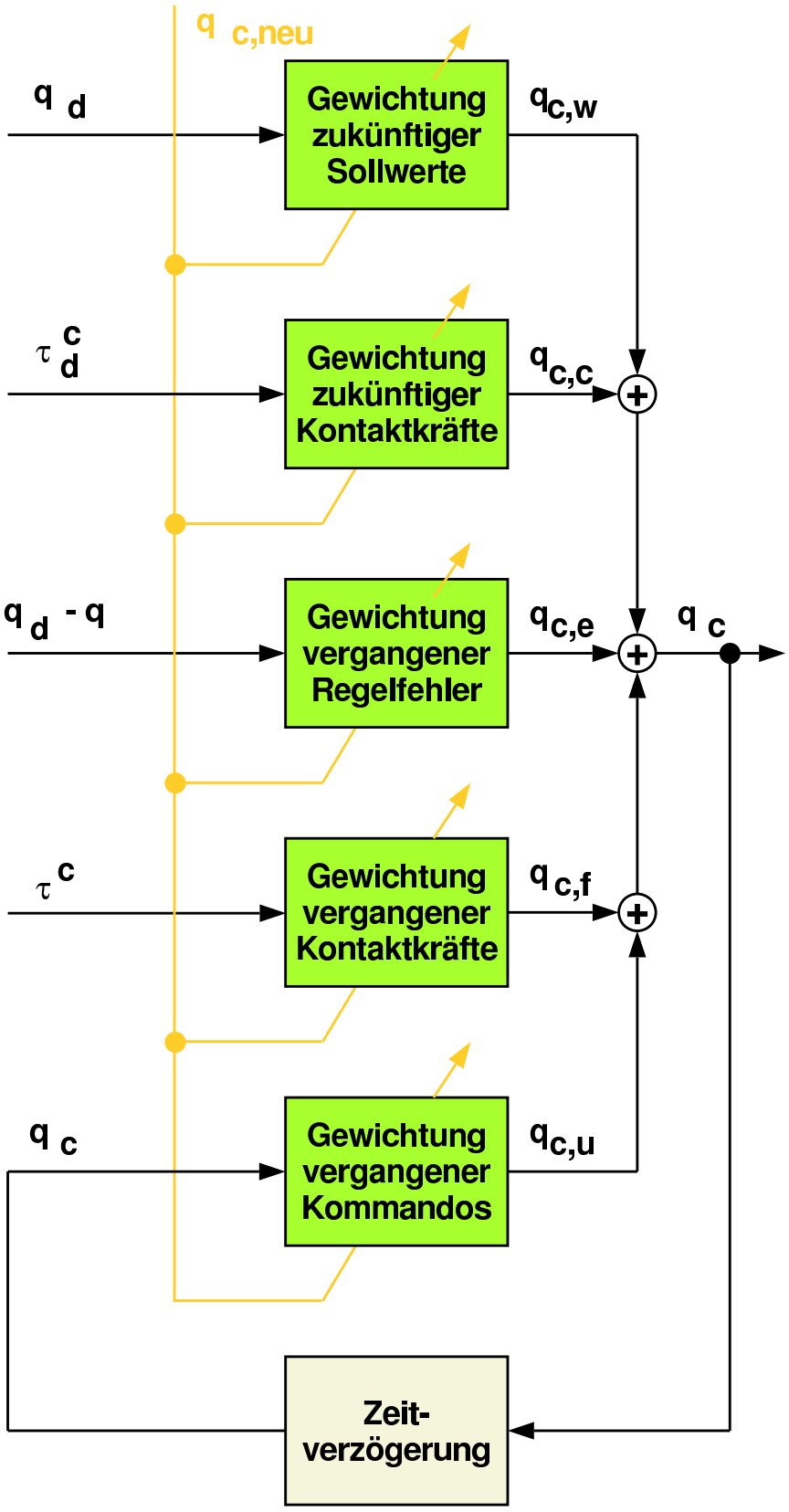


Abbildung 5.10: Struktur der zusätzlich zur internen Kaskadenregelung vorgeschlagenen linearen Reglerelemente (hell gezeichnet ist die Adaption aufgrund der optimierten Stellfolge; die Nullpunktverschiebung um $q_d(k)$ bzw. $\tau_d^c(k)$ ist nicht dargestellt)

Regelung aufgrund der veränderten Position genauso schnell wie die mögliche Kraftrückführung. Der Vorteil der Kraftmessung, dass zwischen stochastischen und systematischen Störungen unterschieden werden kann, wird durch die höhere Bandbreite der Lageregler ausgeglichen.

Der Ablauf des Trainings mit Berücksichtigung von Kräften erfolgt auch nach Abb. 5.6 bzw. 5.9. Lediglich die Schätzung des Vorfilters bzw. des Reglers ist durch die zusätzlichen Parameter umfangreicher. Unter Umständen kann man das Training aber teilen, indem zuerst, bei Trainingsbahnen ohne Kontakt, nur der Positionsteil adaptiert wird, während danach, bei Bahnen mit Krafteinwirkung, die Elemente r_{ci} und r_{fi} adaptiert werden, unter Beibehaltung der r_{wi}, r_{ei} und r_{ui}. Durch das zweigeteilte Training ist die Vorsteuerung robuster gegenüber untrainierten Bahnen, bei denen größere oder kleinere Kräfte wirken. Das kommt daher, dass Kräfte und Positionen bei glatten Kontaktflächen relativ stark korreliert sind, sodass die Parameterschätzung die Einflüsse nur schwer zuordnen kann.

Insgesamt lässt sich erwarten, dass eine gute Verallgemeinerbarkeit auf untrainierte Trajektorien und / oder Störungen der Roboterbewegung gegeben ist. Sofern ein Ansatz nach Gleichung (5.54) mit einer genügend großen Zahl an Parametern nicht in der Lage ist, die durch die zweite Adaptionsstufe vorgegebenen Stellfolgen abzubilden oder untrainierte Bahnen geeignet zu regeln, muss ein nichtlinearer Ansatz oder zumindest eine nichtlineare Ergänzung des linearen Ansatzes erfolgen. Dies ist im nächsten Abschnitt dargestellt.

Ein anderer Grund für schlechte Darstellbarkeit oder mangelnde Verallgemeinerbarkeit kann in einer inkonsistenten Optimierungsstufe liegen, z. B. durch zu klein gewähltes n_o. Dies wurde in Abschnitt 5.2 bereits diskutiert und durch geeignete Maßnahmen ausgeschlossen.

Bei einer konsistenten Stellfolge als Sollwerte zum Training der Parameter der linearen Elemente ist auch die Konvergenz gesichert. Da das Training der bisher beschriebenen Elemente im Gegensatz zum Training Neuronaler Netze keine Iteration benötigt, besteht grundsätzlich kein Konvergenzproblem innerhalb eines Aufrufs der Regleradaption. Bei konsistenten Sollwerten können darüber hinaus große Schwankungen der Parameter zwischen zwei Adpationsschritten ausgeschlossen werden.

5.5 Repräsentation und Adaption nichtlinearer Eigenschaften

In Kapitel 3 wurde festgestellt, dass Industrieroboter sich in erster Näherung linear verhalten. Dementsprechend enthalten die Abschnitte 5.1 bis 5.4 nur lineare Elemente. Dieser Abschnitt beschreibt eine mögliche erweiterte Darstellung. In Kapitel 6 zeigt sich dann, in welchen Fällen dadurch die Bahntreue gegenüber dem linearen Ansatz verbessert werden kann.

5.5.1 Nichtlinearitäten aufgrund von ortsabhängigem Verhalten

Die Modellgleichung (5.17) und die Reglergleichung (5.47) wurden unter der Annahme formuliert, dass sich das Verhalten des Roboters linear beschreiben lässt. Am Beispiel der Modellgleichung soll nun erläutert werden, wie eine Ortsabhängigkeit der Roboterdynamik berücksichtigt werden kann.

Die Dynamik eines Robotergelenks enthält sowohl die Trägheit der Motorbewegung als auch die Trägheit der Armbewegung. Dabei wirkt die Motorbewegung aufgrund der großen Übersetzung im Getriebe weit stärker, als aufgrund des geringen Trägheitsmoments zu vermuten ist. Das wirksame Trägheitsmoment ergibt sich durch Multiplikation des Trägheitsmoments des Rotors mit dem Quadrat des Übersetzungsverhältnisses. Bei heutigen Industrierobotern kommt daher die dynamische Verzögerung weitgehend durch die Beschleunigung der Motoren. Diese Beschleunigung ist unabhängig von den Achsstellungen, wodurch sich eine weitgehende Ortsunabhängigkeit der Dynamik ergibt. Es gibt jedoch Bauformen von Robotern, bei denen die Übersetzungsverhältnisse so klein sind, dass die Beschleunigungen der Roboterarmelemente merkbar werden. Die wirksamen Trägheitsmomente der Armelemente sind zumindest bei den ersten Gelenken abhängig von den Stellungen der weiteren Gelenke. Das bedeutet eine Ortsabhängigkeit der Dynamik. Dabei ist mit „Ort“ bei einem Roboter mit 6 Gelenken seine Position im Gelenkraum gemeint, also nicht nur die kartesische Position des Endeffektors.

Die Modellgleichung (5.17) mit ortsabhängigen Parametern lautet:

$$
\begin{aligned}
y(k+1) = u(k) \;&+\; \hat{g}_1(\mathbf{x}(k)) \cdot (u(k-n_t) - u(k)) \\
&+\; \hat{g}_2(\mathbf{x}(k)) \cdot (u(k-n_t-1) - u(k)) \\
&+\; \hat{g}_3(\mathbf{x}(k)) \cdot (u(k-n_t-2) - u(k)) \\
&+\; \cdots \\
&+\; \hat{g}_{n_m}(\mathbf{x}(k)) \cdot (u(k-n_t-n_m+1) - u(k))
\end{aligned}
\tag{5.55}
$$

Dabei werden, wie bei Gleichung (5.17), die Bewegungsanweisungen von n_m Abtastschritten gewichtet. $\mathbf{x}(k)$ beschreibt den „Zustand“ zum Zeitpunkt k, wobei dieser bei reiner Ortsabhängigkeit nur aus den Achswerten besteht. Dabei kann die Zahl der Achswerte in Abhängigkeit der Kinematik eingeschränkt werden. So haben z. B. bei den beiden untersuchten Robotern die absolute Stellung des ersten und letzten Gelenks keinen Einfluss auf die Dynamik. $\mathbf{x}$ hat dort also nur 4 Komponenten.

Diese parametrische Form ist wesentlich günstiger als die allgemeine nichtlineare Darstellung

$$
\begin{aligned}
y(k+1) = u(k) \;&+\; f(\mathbf{x}(k), u(k), \\
&\qquad u(k-n_t), u(k-n_t-1), \cdots, u(k-n_t-n_m+1)),
\end{aligned}
\tag{5.56}
$$

da die Funktion f bei $n_m = 14$[15] und einem Vektor $\mathbf{x}$ mit 4 Freiheitsgraden zusammen mit $u(k)$ insgesamt 19 skalare Eingänge hat, was unter der Annahme linearer Unabhängigkeit der Abbildung eines 19-dimensionales Raum entspricht. Das ist nur mit großen Aufwand möglich. Dagegen werden bei Gleichung (5.55) Funktionen beschrieben, die jeweils nur von 4 Eingangsgrößen abhängen. Wegen der exponentiellen Abhängigkeit des Aufwands von der Dimension des abzubildenden Raumes ist es wesentlich günstiger, 14 Funktionen mit jeweils 4 Eingangsgrößen abzubilden als eine Funktion mit 19 Variablen.

Die Darstellung nach Gleichung (5.55) ist außerdem für die Anwendung des Modells im zweiten Schritt der Adaption günstiger, da dort eine parametrische Form benötigt wird. So können die Werte der Funktionen $\hat{g}_i$ direkt in Gleichung (5.23) oder Gleichung (5.35) eingesetzt werden. Dabei werden zeilenweise die

[15] $n_m = 14$ wurde auf Seite 132 für einen Manutec r2 Roboter bestimmt. Bei anderen Robotertypen muss n_m entsprechend gewählt werden.

Soll-Gelenkwinkel $w_i(k)$ in den Roboterzustand $\mathbf{x}(k)$ eingesetzt, um die $\hat{g}_j(\mathbf{x}(k))$ zu bestimmen.

Für Darstellungen entsprechend Gleichung (5.55) wurde der Tabellenspeicher ME9 enwickelt [148], der Funktionen mit einigen nichtlinear wirkenden Eingängen wie dem Roboterzustand und vielen linear wirkenden Eingängen wie den Stellgrößendifferenzen darstellt. Diese Repräsentation des gelernten Wissens wurde z. B. in [147] für einen ortsabhängigen Regler eingesetzt.

Die nichtlinearen Funktionen werden bei ME9 ähnlich der Identifikation der Modellparameter in Abschnitt 5.1 geschätzt. Dabei wird für jeden Punkt des von den nichtlinear wirkenden Variablen aufgespannten Zustandsraumes eine getrennte Schätzung der Parameter einer internen polynomiellen Darstellung vorgenommen. Bei quadratischem Ansatz und zwei nichtlinear wirkenden Variablen $\mathbf{x} = (x_1, x_2)^T$ ergibt sich die Schätzung der Parameter $\theta_{ij}(\mathbf{x}_s)$ eines Punktes $\mathbf{x_s}$ aufgrund eines Sollwertes $y(k+1)$ und der dazugehörigen linear wirkenden Eingangswerte $u(k-i) - u(k)$ analog zu Gleichung (5.55) durch

$$
\begin{aligned}
y(k+1) \;=\;& u(k) \\
&+\; (\theta_{11}(\mathbf{x_s}) + \theta_{12}(\mathbf{x_s}) \cdot (x_1(k) - x_{s1}) + \theta_{13}(\mathbf{x_s}) \cdot (x_2(k) - x_{s2}) \\
&\quad +\theta_{14}(\mathbf{x_s}) \cdot (x_1(k) - x_{s1}) \cdot (x_2(k) - x_{s2}) \\
&\quad +\theta_{15}(\mathbf{x_s}) \cdot (x_1(k) - x_{s1})^2 + \theta_{16}(\mathbf{x_s}) \cdot (x_2(k) - x_{s2})^2) \\
&\qquad \cdot (u(k - n_t) - u(k)) \\
&+\; (\theta_{21}(\mathbf{x_s}) + \theta_{22}(\mathbf{x_s}) \cdot (x_1(k) - x_{s1}) + \theta_{23}(\mathbf{x_s}) \cdot (x_2(k) - x_{s2}) \\
&\quad +\theta_{24}(\mathbf{x_s}) \cdot (x_1(k) - x_{s1}) \cdot (x_2(k) - x_{s2}) \\
&\quad +\theta_{25}(\mathbf{x_s}) \cdot (x_1(k) - x_{s1})^2 + \theta_{26}(\mathbf{x_s}) \cdot (x_2(k) - x_{s2})^2) \\
&\qquad \cdot (u(k - n_t - 1) - u(k)) \\
&+\; \cdots
\end{aligned}
\tag{5.57}
$$

Die Schätzung erfolgt durch ein rekursives Kalman Filter[16], bei dem die angenommene Störung σ^2 in Abhängigkeit von $||\mathbf{x}(k) - \mathbf{x}_s||$ für jeden Punkt $\mathbf{x_s}$ und jeden Trainingsschritt k individuell angepasst wird. Dadurch ergibt sich eine trainingsabhängige Verallgemeinerung, bei der die Messwerte bei geringen Informationen im Rahmen des Ansatzes durch Gleichung (5.57) fast glo-

[16] Auch hier wird nur die Messgleichung betrachtet, sodass das Kalman Filter als spezielle Form eines Least-Square-Algorithmusses gesehen wird, ohne Berücksichtigung einer Dynamikgleichung (Systemgleichung).

bale Gültigkeit haben, während bei weiterem Training die Verallgemeinerung eingeschränkt wird, um auch die Repräsentation lokal begrenzter Einflüsse zu ermöglichen.

Die bisherigen Anwendungen des Tabellenspeichers beschränken sich auf ein oder zwei nichtlinear und 6 linear wirkende Variablen. Dabei werden jeweils $(1+1+1)\cdot 6 = 18$ bzw. $(1+2+3)\cdot 6 = 36$ Parameter geschätzt. Eine starke Erhöhung der Zahl der zur internen Darstellung nötigen Parameter ist problematisch, da die Schätzung dann sehr empfindlich gegenüber Störungen ist. So ist der Tabellenspeicher für das oben erwähnte Beispiel mit 4 nichtlinear und 14 linear wirkenden Parametern in der Praxis nicht mehr anwendbar, da er bei quadratischem Ansatz jeweils $(1+4+10)\cdot 14 = 210$ Variablen bestimmen müsste. Da es auf der anderen Seite kein so gut an die Modellstruktur angepasstes Verfahren zur Repräsentation hochdimensionaler Abbildungen gibt, muss versucht werden, die Zahl der Variablen in Abhängigkeit der speziellen Roboterkinematik möglichst stark einzuschränken.

Dabei wird ausgenutzt, dass das wirksame Trägheitsmoment eines Armelements nur von den Stellungen der weiteren Gelenke abhängig ist. Für die in dieser Arbeit untersuchten Roboter ist darüber hinaus aufgrund der Symmetrie des Endeffektors die Dynamik von Gelenk 6 und Gelenk 5 nicht ortsabhängig, die Dynamik von Gelenk 4 hängt nur von der Stellung von Gelenk 5 ab und auf Gelenk 3 haben nur Gelenk 4 und 5 Einfluss. Bei Gelenk 1 und 2 wirkt hauptsächlich die Stellung von Gelenk 2 und 3 bzw. nur von Gelenk 3, da die Handachsen geringere Veränderungen der Trägheitsmomente bewirken. Die Zahl der nichtlinear wirkenden Eingänge lässt sich also bei den untersuchten Robotern auf zwei begrenzen. Ähnliche Überlegungen lassen sich auch bei den meisten anderen Industrierobotertypen anstellen. Die Zahl der linear wirkenden Eingänge lässt sich von n_m auf n_g halbieren, da die Erweiterung von $n_g = 7$ auf $n_m = 14$ nur zur Abschätzung des durch diese Annahme verursachten Fehlers dient (vergl. Abschnitt 5.1).[17]

Damit ist die Identifikation einer ortsabhängigen Modelldarstellung für den untersuchten Manutec r2 Roboter möglich. Für jeden Schätzpunkt $\mathbf{x}_s$ werden für jeden Ausgang, also für jede Achse des Roboters, $(1+2+3)\cdot 7 = 42$ Schätzwerte $\theta_{11}(\mathbf{x}_s)\cdots\theta_{76}(\mathbf{x}_s)$ geschätzt. Die Repräsentation nichtlinearer Vorsteuerungen oder Regelungen erfolgt in gleicher Weise.

[17] Die Zahlenwerte gelten nur für den in Abschnitt 5.1 untersuchten Roboter.

Ähnliche Ansätze zur Darstellung nichtlinearer Abbildungen durch mehrere örtlich begrenzt gültige lineare Repräsentationen sind in der Literatur zu finden (z. B. Nelles [186]). Dabei wird das Training i. A. mit einem global gültigen linearen Modell begonnen, dessen Gültligkeitsbereich halbiert wird, sofern Widersprüche unter den Trainingsdaten auftreten.

5.5.2 Neuronale Netze zum Abbilden der nichtlinearen Abhängigkeiten

Im Kontext dieser Arbeit lassen sich Neuronale Netze prinzipiell bei beiden Wissensbasen einsetzen, also beim Modell und beim Regler. Dabei können neben Nichtlinearitäten der einzelnen Gelenkfunktionen auch Kopplungen zwischen den Teilsystemen nichtlinear abgebildet werden.

Die Implementierung des Reglers durch ein Neuronales Netz (siehe auch Abb. 5.4) ist unproblematisch, da dort nur der Funktionswert ausgelesen wird. Bei Realisierung des Modells durch ein Netz muss dagegen die Optimierung angepasst werden, da dort die Parameter der internen Funktionsdarstellung verwendet werden. Deshalb ist dort das in Abschnitt 5.5.1 beschriebene Verfahren vorzuziehen. Wie die Experimente in Kapitel 6 im Gegensatz zu den in der Literatur beschriebenen Verfahren beweisen, reicht aber ein lineares Modell aus.

Grundsätzlich muss bei iterativer Adaption das Modell nicht so genau sein wie der Regler. Es reicht, wenn das Modell die Richtung einer Korrektur der Reglerparameter angeben kann und wenn die gewählte Schrittweite nicht zu Schwingungen bei der Adaption führt. Dagegen ist beim Regler auch der Betrag einer Stellgröße von Bedeutung, sofern die Regelung nicht nur stabil, sondern auch optimal bezüglich eines Gütemaßes sein soll.

In diesem Abschnitt werden daher nur Nichtlinearitäten bezüglich der in Abschnitt 5.4 beschrieben Eingangsvariablen der Vorsteuerung bzw. Regelung behandelt. Dazu wurde in Abb. 5.4 eine parallele Struktur aus linearen Elementen und Neuronalen Netzen vorgeschlagen. Bei Letzteren werden in dieser Arbeit insbesondere mehrschichtige Perzeptrons mit differenzierbarer Aktivierungsfunktion und Netze mit radialen Basisfunktionen (RBF-Netzwerke) [198] betrachtet.

Zur Vereinfachung des (überwachten) Trainings wird für jede zu trainierende Funktion ein Neuronales Netz und eine lineare Darstellung kombiniert, da die lineare Schätzung durch das Kalman Filter sehr schnell konvergiert. Für das Neuronale Netz bleibt somit nur der nichtlineare (verkoppelte) Teil der Funktion. Dadurch ist die relative Repräsentationsgenauigkeit im Gesamtsystem höher als die durch ein Netz allein erreichbare. Es kann sogar eine obere Fehlerschranke angegeben werden, nämlich der Darstellungsfehler des linearen Systems allein.

Diese Kombination aus vorgegebener Struktur und Neuronalem Netz kann als Verwendung von a priori Kenntnissen interpretiert werden. Ähnliche Ansätze sind auch in der Literatur zu finden. So werden bei Katic und Vukobratovic [121] ein ungenaues vorgegebenes inverses Robotermodell und der daraus gebildete Regler jeweils mit einem Neuronalen Netz kombiniert. Bei de Angulo und Torras [60] werden anstelle der kinematischen Transformation im Netz nur die Abweichungen gegenüber der angenommenen Kinematik trainiert. Bei Licheng et al. [161] wird ein modellbasierter Regler mit einem Neuronalen Netz kombiniert, wobei letzteres sich auf die Regelung eines flexiblen Armelements innerhalb des sonst starren Roboters beschränkt. Auch der Tabellenspeicher in [148] benutzt, soweit möglich, a priori bekannte Strukturkenntnis, indem er zwischen vielen linear und wenigen nichtlinear wirkenden Eingängen unterscheidet, um eine „arbeitspunktabhängige lineare Darstellung“ zu erzeugen. Ein weiteres Beispiel für die Nutzung von a priori Kenntnissen ist die Einschränkung der Zahl der Netzeingänge[18], die auf der Kinematik des Roboters beruht. Darauf wird im Abschnitt 5.5.3 eingegangen.

Bezüglich der Verallgemeinerung, also der Fähigkeit, für nicht trainierte Zwischenwerte interpolierte Ausgangswerte zu erzeugen, gilt das bisher Gesagte entsprechend. Auch hier ist die lineare Darstellung bei linearen Funktionen genau lernbar und benötigt ein Mindestmaß an Trainingsinformation. Im Gegensatz dazu ist die Verallgemeinerung bei Neuronalen Netzen stark struktur- und trainingsabhängig. Die Kombination der linearen und der nichtlinearen Darstellung verspricht auch hier die Einhaltung einer oberen Fehlerschranke, selbst bei spärlichem Training.

[18] Grundsätzlich muss festgestellt werden, dass eine Funktion nur dann trainierbar ist, wenn die verkoppelten physikalischen Einflüsse wenige Freiheitsgrade haben. Dagegen darf die Zahl der Eingangsgrößen prinzipiell sehr groß sein. Der im Abschnitt 5.5.3 vorgeschlagene Eingangsraum mit 14 bzw. 24 Variablen stellt allerdings so hohe Anforderungen, dass von noch höher dimensionalen Eingangsräumen und der damit verbundenen Zahl an Gewichten abgeraten werden muss, besonders wenn neben der Reproduktion der Trainingsdaten auch noch verallgemeinert werden soll. Daher ist es sinnvoll, auch die Zahl der Eingänge auf das Mindestmaß zu reduzieren.

Da der Ausgang eines Neuronalen Netzes wertmäßig begrenzt ist, muss der Ausgangswert durch geeignete Skalierung an den gewünschten Sollwert angepasst werden. Durch die parallele Struktur aus linearem und nichtlinearem Anteil kann die Skalierung klein gewählt werden. Dadurch wird der Anteil des Netzes an der gesamten Funktion auf z. B. 10 % des Maximalwertes beschränkt. Dies ist dann bei untrainierten Werten und ungünstiger Verallgemeinerung auch die maximal mögliche Verschlechterung durch das Netz.

Neben diesen maximalen Fehlern sollen auch die wahrscheinlichen Fehler diskutiert werden. Der lineare Teil der Darstellung kann, sofern die Zahl der Trainingspunkte mindestens gleich der Zahl der Parameter ist, ideal interpolieren und extrapolieren. Zusammen mit dem nichtlinearen Anteil wird in dem von den Trainingspunkten eingeschlossenen konvexen Raum bei ausreichendem Training und geeigneter Netzstruktur sicherlich eine gegenüber dem linearen Ansatz genauere Darstellung erreicht werden (Interpolation zwischen den Trainingspunkten). Außerhalb dieses Raumes kann aber die im vorigen Absatz berechnete Verschlechterung auftreten, da dort keine Information zur Extrapolation der Nichtlinearitäten vorliegt.

Klare Regeln, welcher Netztyp vorzuziehen ist oder welche Konfiguration empfehlenswert ist, liegen derzeitig nicht vor. Aussagen hierzu sind jeweils abhängig von dem betrachteten Robotertyp, von der Art der zu berücksichtigenden Nichtlinearität (Sättigung der Antriebe, kinematische Kopplungen, Zentrifugalkräfte, usw.) und von der Art des Trainings. Bei dem in dieser Arbeit untersuchten Manutec r2 Roboter sind insbesondere Kopplungen zwischen den Handachsen durch Neuronale Netze abzubilden. Das Training erfolgt offline, weswegen auch ein hoher Trainingsaufwand zu Gunsten einer höheren Güte der Repräsentation in Kauf genommen werden kann.

Für diese Arbeit wird als Netztyp ein mehrschichtiges Perzeptron mit der Aktivierungsfunkton (2.7) ausgewählt. Die genaue Konfiguration wird für das Beispiel der Entkopplung in Abschnitt 5.5.3 beschrieben. Es ist davon auszugehen, dass andere Netztypen, z. B. Netze mit radialen Basisfunktionen (RBF-Netzwerke) die Aufgabe der nichtlinearen Abbildung in gleicher Weise erfüllen können. Dagegen werden CMAC-Netzwerke trotz leichterer Trainierbarkeit [231] nicht empfohlen, da ihr Aufwand bei der Berechnung der Ausgangswerte relativ hoch ist.

Ohne Verlust an Allgemeinheit kann man sich auf Netze mit je einem Ausgang

beschränken. Mehrere kleine Netze mit je einem Ausgang erlauben in der Regel im Vergleich zu einem großen Netz für mehrere Ausgänge ein schnelleres Training. Daher wird für jede darzustellende Funktion ein eigenes Neuronales Netz angesetzt.

Experimentell erweist sich eine Struktur mit zwei verdeckten Schichten als brauchbar. Die Zahl der Neuronen in jeder Schicht wird nach einer auf ganze Zahlen gerundeten geometrischen Reihe von der vorgegebenen Zahl an Eingängen bis zur Ausgangsschicht mit einem Neuron angesetzt. Eine höhere Zahl von Neuronen oder Schichten erfordert mehr Trainingsdaten bzw. erzeugt eine schlechtere Verallgemeinerungsfähigkeit, bei einer geringeren Zahl lässt sich ggf. der Trainingsfehler nicht genügend reduzieren.

Eine problemangepasste Wahl der Struktur ist möglich, indem man zuerst mehr Neuronen wählt und dann die redundanten Verbindungen streicht (Pruning) (siehe z. B. [83, 214, 227]). Dabei sollte man aber einen Teil der Trainingsdaten zur späteren Kontrolle der Verallgemeinerung nicht trainieren.

Bei der in Abschnitt 5.3 empfohlenden parallelen Struktur von linearen Abbildungen und Neuronalen Netzen wird zuerst der lineare Teil der Darstellung adaptiert. Der verbleibende Repräsentationsfehler wird dann als nichtlinearer Anteil der zu trainierenden Funktion angesehen und im Netz trainiert.

Beim Netztraining geht man üblicherweise von stochastisch initialisierten Gewichten oder speziell gelernten Anfangsgewichten [90] aus. Hier sollen dagegen die Gewichte der Ausgangsschicht mit null vorgewählt sein. Diese Vorgabe der Ausgangsgewichte führt bei symmetrischer Skalierung des Ausgangs dazu, dass sich das System mit untrainiertem Netz so verhält, als wäre nur der lineare Teil aktiv. Aufgrund der Konvergenz des Trainingsverfahrens kann der Fehler beim Training dann nur noch abnehmen. Dagegen ergeben anders initialisierte Gewichte zu Beginn des Netztrainings eine Verschlechterung gegenüber der rein linearen Darstellung.

Da das in der Literatur übliche Back-Propagation Verfahren zum Training des Netzes nur sehr langsam konvergiert, wird mit einem erweiterten Kalman Filter (siehe Anhang A.3) trainiert. Die dazu notwendige Anpassung der Trainingsdaten wird im Anhang B entwickelt.

Vorteil des als EKFNet veröffentlichten Verfahrens [151] ist, dass durch eine Kovarianzmatrix die Beziehungen zwischen den einzelnen Trainingsmustern impli-

zit gespeichert werden, so dass durch ein neues Trainingsmuster die Antwort auf die bisherigen Trainingsmuster nur minimal verschlechtert wird. Im Vergleich zum Back-Propagation, bei dem durch eine Lernrate und einen Momentum-Term eine heuristische Schrittweitensteuerung erfolgt, wird bei EKFNet die zur Steuerung der Adaption verwendete Kalman Verstärkung durch den Algorithmus selbst angepasst. Der im Gegensatz zum Back-Propagation quadratische Aufwand in n (durch die $(n \times n)$ Kovarianzmatrix) ist bei Approximationsaufgaben bis zu etwa $n = 300$ Parametern akzeptabel, da nur wenige Iterationen zum Netztraining notwendig sind (vergl. Abschnitt 6.2).

Andere Gradientenverfahren (z. B. Levenberg-Marquardt, Truncated Newton, Quasi-Newton, Conjugate Gradient (= Spezialfall von Back-Propagation mit Momentum), Conjugate gradient with Powell restarts) [39, 58, 227, 247, 258] wurden auch untersucht, versprechen aber kein besseres Verhalten bzw. erwiesen sich im Experiment als weniger geeignet.

Um Missverständnissen vorzubeugen soll noch einmal betont werden, dass es sich bei dem Training der Netze in allen Fällen um überwachtes Lernen handelt. Dabei sind die Sollwerte durch die zweite Stufe der Adaption, also durch die Optimierung der Stellgrößen vorgegeben. Es ist also kein Reinforcement-Learning oder eine andere Form des Lernens unter Beobachtung des Prozessverhaltens nötig.

Dadurch ist prinzipiell die Konvergenz des Netztrainings gesichert, sofern ein geeignetes Trainingsverfahren verwendet wird. Dies ist bei EKFNet der Fall. Bei normalisierten Netzen (Netzen für Ein- und Ausgangsfunktionen in der Größenordnung von $(0, 1)$) müssen dabei auch keine Lernparameter vorgegeben werden, andernfalls lassen sich die Lernparameter leicht anpassen. Ob das fertige Netz die Trainingsinformation fehlerfrei repräsentieren kann, ist dadurch allerdings noch nicht garantiert. Dies ist u. a. von den gewählten Netzeingängen abhängig.

Bei den zur Verfügung stehenden Robotern wurde nichtlineares Verhalten insbesondere in Form von Kopplungen beobachtet. Daher beschäftigt sich Abschnitt 5.5.3 näher mit dieser Problematik.

5.5.3 Kompensation von Kopplungen durch Neuronale Netze

Bevor hier auf den konkreten Einsatz der Neuronalen Netze eingegangen wird, soll das Problem zunächst grundlegend beschrieben werden.

In Abschnitt 5.4 wird angenommen, dass die dynamische Beschreibung der einzelnen Achsen linear und entkoppelt ist. Dies ist in der Praxis sicherlich nicht der Fall. Experimente mit einem Manutec r2 Roboter zeigen, dass die Kopplungen einen größeren Einfluss haben als die Nichtlinearitäten der achsweisen Beschreibungen. Dabei wurde eine ortsabhängige Vorsteuerung mit dem Tabellenspeicher [148] implementiert, die aber nicht die Verbesserung der in Kapitel 6 mit Neuronalen Netzen erreichten Entkopplung erreichte. Das lässt sich dadurch erklären, dass die ersten Gelenke, die am meisten von ortsabhängig schwankenden Trägheitsmomenten gestört sein müssten, bei den meisten Robotern im Gegensatz zu den Handachsen allein durch lineare Vorsteuerung schon recht genau werden. Dagegen haben das 4. und 5. Gelenk bei dem untersuchten Manutec r2 Roboter große Abweichungen, obwohl sich bei diesen Achsen, abgesehen von der Kopplung, kaum ein ortsabhängiger Einfluss erkennen lässt.

Es zeigt sich weiterhin, dass die Messwertrückführung keine Kopplungen bewirkt, da sie i. A. nur kleine Abweichungen von den nominal optimalen Stellgrößen erzeugt. Somit bleibt für eine nichtlineare Darstellung nur die Vorsteuerung. Es geht also darum, aufgrund der Kopplung durch andere Gelenke, die beim Ansatz nach Abschnitt 5.4 nicht berücksichtigt werden kann, eine nichtlineare Vorsteuerung aufzubauen, ähnlich wie die lineare Vorsteuerung zur Kompensation der Trägheit des betrachteten Gelenks selbst.

Dazu eignet sich prinzipiell ein analytisches Modell, bei dem die Parameter der Vorsteuerung trainiert werden. Hier soll dagegen der allgemeine Fall behandelt werden, bei dem weder Parameter noch Struktur vorgegeben werden. Das Netz soll also die Kopplungen nur aufgrund der Trainingsdaten und der nach Abschnitt 5.2 geschätzten optimalen Stellgrößen lernen. Dementsprechend bleibt in diesem Abschnitt auch das adaptive System in der beschriebenen Form erhalten. Lediglich die Vorsteuerung wird um Neuronale Netze erweitert, die parallel zur linearen Abbildung einen Teil zu den Stellgrößen beitragen.

Es ist also nötig, die nichtlineare Vorsteuerung einer Achse auch von den Werten der anderen Achsen abhängig zu machen. Dabei haben neben den Achsstellungen, die die Geometrie beschreiben, auch die Achsgeschwindigkeiten und

-beschleunigungen Einfluss.

Zur Kompensation der Kopplung reicht es nicht, nur die Koppelterme zum jeweiligen Zeitpunkt zu betrachten. Stattdessen muss man für die Koppelgrößen Vorfilter einführen, die anders als die bisher bestimmten sind, da die Dynamik der verschiedenen Gelenke unterschiedlich ist. So wird die Kopplung der Beschleunigungen von Gelenk 1 auf Gelenk 4 ein Vorfilter ergeben, das die aufgrund des Vorfilters für Gelenk 1 ermittelten Bewegungsanweisungen mit der Dynamik von Gelenk 4 kombiniert.

Zählt man nun die Variablen zusammen, die Einfluss auf ein einzelnes Gelenk ausüben können, so stellt man fest, dass ihre Zahl so groß ist, dass sich die Adaption zumindest schwierig, wenn nicht unmöglich gestaltet. Das gilt auch dann, wenn die Struktur der Kopplungen durch ein genaues Robotermodell bekannt ist, bei dem lediglich die Roboterparameter bestimmt werden müssen. Man muss daher abschätzen, welche Kopplungen überhaupt wesentlich sind. Dementsprechend ist, speziell für jedes Gelenk, eine Reduktion der Zahl der Eingangsgrößen möglich.

Untersuchungen an einem Manutec r2 Roboter zeigen, dass die ersten drei Gelenke aufgrund der großen Trägheitsmomente und der stärkeren Antriebe weniger von Kopplungen betroffen sind als das 4. und 5. Gelenk. Das 6. Gelenk reagiert wiederum wegen des symmetrischen Endeffektors kaum auf die übrigen. Es scheint also ausreichend, Kopplungen nur bei dem 4. und 5. Gelenk zu berücksichtigen. Diese Annahme wird durch die Experimente in Kapitel 6 bestätigt.

Dabei zeigt sich, wie die Analyse zu Abb. 6.2 zumindest für das 4. Gelenk beweist, dass die Fehler vorwiegend durch Beschleunigungen der anderen Gelenke entstehen, nicht aber aufgrund von geschwindigkeitsabhängigen Zentrifugal- oder Corioliskräften. Um dies zu bestätigen, wird ein analytisches Robotermodell

$$\mathbf{M}(\mathbf{q}) \cdot \ddot{\mathbf{q}} + \mathbf{V}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{G}(\mathbf{q}) = \tau \tag{5.58}$$

angesetzt. Dabei sind $\mathbf{q}$ die Gelenkwinkel, $\mathbf{M}$ ist die Massenmatrix, $\mathbf{V}$ beschreibt die Zentrifugal- und Coriolisterme und $\mathbf{G}$ enthält die Gravitationsabhängig-

keit[19]. τ ist der Vektor der kommandierten Gelenkmomente.

Für die i-te Zeile von Gleichung (5.58) ergibt sich

$$m_{ii}(\mathbf{q}) \cdot \ddot{q}_i = \tau_i - v_i(\mathbf{q}, \dot{\mathbf{q}}) - g_i(\mathbf{q}) - \sum_{i \neq j} m_{ij}(\mathbf{q}) \cdot \ddot{q}_j. \tag{5.59}$$

Durch die relativ großen Rotorträgheitsmomente kann man die Diagonalelemente der Massenmatrix als dominant gegenüber den übrigen Elementen annehmen. Das rechtfertigt den Ansatz der Robotersteuerung, die Gelenke entkoppelt zu betrachten und einzeln zu regeln.

Dynamische Kopplungen ergeben sich also durch Zentrifugal- oder Coriolisterme in $\mathbf{V}(\mathbf{q}, \dot{\mathbf{q}})$ und durch die Nebendiagonalelemente der Massenmatrix. Die statischen Einflüsse der Gravitationsmatrix wirken sich dagegen wegen der unterlagerten Regelung nicht aus. Im Gegensatz sind die beschleunigungsabhängigen Terme besonders unangenehm, da sie sich sehr schnell ändern können.

Bei einer analytischen Formulierung werden sowohl Zentrifugal- als auch Coriolisbeschleunigungen durch einen die Geometrie beschreibenden Vorfaktor und das Produkt aus zwei Gelenkgeschwindigkeiten beschrieben. Die Nebendiagonalelemente der Massenmatrix enthalten ähnliche Geometriebeschreibungen und werden mit einer Gelenkbeschleunigung multiplitiziert. Daher lässt sich durch die maximal vorkommenden Gelenkgeschwindigkeiten und -beschleunigungen nach Tabelle 5.1 der Einfluss dieser Ausdrücke abschätzen.

Tabelle 5.1 zeigt, dass die im Betrieb maximal erreichten Beschleunigungen um eine Größenordnung über den Geschwindigkeiten liegen und damit eher der Grund für Kopplungen sind. Damit wird die Vermutung aufgrund der Tests mit dem 4. Gelenk (Abb. 6.2) bestätigt.

Als Eingänge zur Kompensation von Kopplungen durch Beschleunigungen kommen bei Robotern mit einer kinematischen Struktur nach Abb. 3.13 ähnlich wie bei der ortsabhängigen Darstellung in Abschnitt 5.5.1 die Soll-Gelenkpositionen w_2, w_3, w_4 und w_5 in Frage. Zusätzlich sind für das 4. Gelenk die Änderungen von w_1 gegenüber dem aktuellen Wert nötig, um die beschleunigungsabhängigen Kopplungen vom 1. auf das 4. Gelenk zu modellieren. Andere Gelenke haben

[19] Die Bezeichnungen $\mathbf{G}$ und g_i der Gleichungen (5.58) und (5.59) widersprechen der ansonsten in dieser Arbeit verwendeten Notation, die damit die Gewichtsfunktion beschreibt. Sie sind allerdings allgemein üblich, so dass darauf verzichtet wurde, andere, widerspruchsfreie Bezeichnungen einzuführen.

Gelenk	max. Geschwindigkeit	max. Beschleunigung
1	1.6 rad/s	13.8 rad/s^2
2	1.4 rad/s	10.5 rad/s^2
3	2.0 rad/s	12.6 rad/s^2
4	0.0 rad/s	2.1 rad/s^2
5	1.5 rad/s	11.0 rad/s^2
6	1.7 rad/s	19.3 rad/s^2

Tabelle 5.1: Maximale Gelenkgeschwindigkeiten und -beschleunigungen, die beim Abfahren von Bahn 3 mit 500 mm/s gemessen werden (aus den Abtastwerten der Gelenkpositionen berechnet)

kaum Einfluss auf das 4. Gelenk.

Das 5. Gelenk ist auch vom Absolutwert des 1. und 6. Gelenks unabhängig, benötigt aber sicherlich die übrigen Winkel zur Beschreibung der Konfiguration. Dazu kommen wegen der Kopplung durch die Beschleunigungen des 2. und 3. Gelenks deren Differenzen zum aktuellen Wert.

Experimentell wird festgestellt, dass es zur Konfiguration eines Netzes günstiger ist, wenn die Sollwertdifferenzen zur Bestimmung von $u(k)$ ab dem nächsten Schritt $(k+1)$ berücksichtigt werden und nicht erst nach Ablauf der Totzeit, wie bei der linearen Vorsteuerung. Dadurch ergibt sich für die Zahl der jeweils verwendeten Sollwertdifferenzen $n_k = n_w + n_t = 10$.

Zusammen sind das 14 Eingangsvariablen ($w_2(k)$, .. , $w_5(k)$, $(w_1(k+1) - w_1(k))$, .. , $(w_1(k+n_k) - w_1(k))$) für das 4. Gelenk und 24 Eingangsvariablen ($w_2(k)$, .. , $w_5(k)$, $(w_2(k+1) - w_2(k))$, .. , $(w_2(k+n_k) - w_2(k))$, $(w_3(k+1) - w_3(k))$, .. , $(w_3(k+n_k) - w_3(k))$) für das 5. Gelenk, insgesamt 34 Variablen der nichtlinearen Anteile für beide Gelenke. Dabei ist es günstiger, für jedes Gelenk ein Netz zu nehmen, da das Training von zwei Netzen mit 14 und 24 Eingängen wesentlich schneller als das Training eines Netzes mit 34 Eingängen geht. Damit wird der Ansatz von Gleichung (5.44) für die Vorsteuerung von Gelenk 4 und 5 folgendermaßen erweitert:

$$\begin{aligned} u_4(k) &= w_4(k) + r_{w41} \cdot (w_4(k+n_t+1) - w_4(k)) + \cdots \\ &\qquad + r_{w4n_w} \cdot (w_4(k+n_t+n_w) - w_4(k)) \\ &+ f_4(w_2(k), w_3(k), w_4(k), w_5(k), w_1(k+1) - w_1(k), \cdots, \\ &\qquad w_1(k+n_k) - w_1(k)) \end{aligned} \tag{5.60}$$

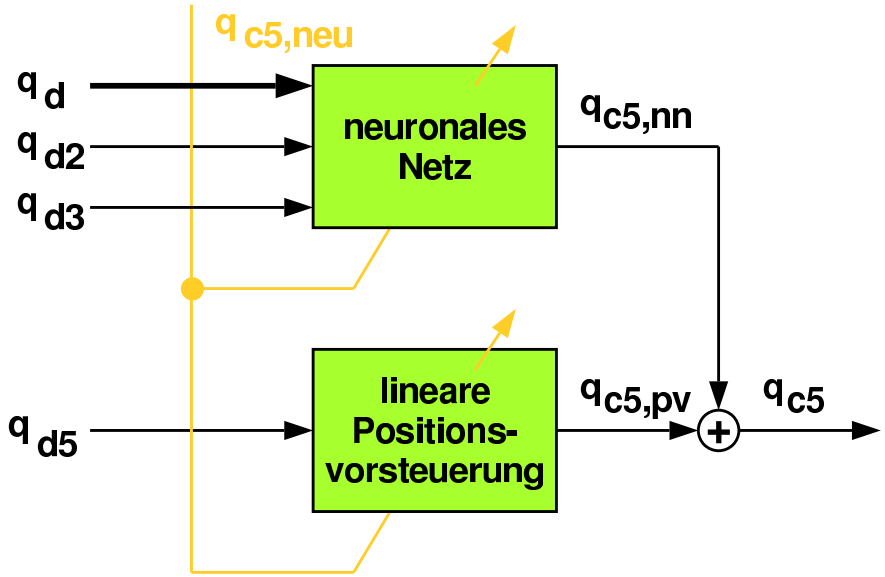


Abbildung 5.11: Struktur der zusätzlich zur internen Kaskadenregelung vorgeschlagenen linearen und nichtlinearen Positionsvorsteuerung für Gelenk 5 (hell gezeichnet ist die Adaption aufgrund der optimierten Stellfolge; die Nullpunktverschiebungen sind nicht dargestellt)

$$\begin{aligned} u_5(k) &= w_5(k) + r_{w51} \cdot (w_5(k+n_t+1) - w_5(k)) + \cdots \\ &\qquad + r_{w5n_w} \cdot (w_5(k+n_t+n_w) - w_5(k)) \\ &+ f_5(w_2(k), w_3(k), w_4(k), w_5(k), w_2(k+1) - w_2(k), \cdots, \\ &\qquad w_2(k+n_k) - w_2(k), w_3(k+1) - w_3(k), \cdots, \\ &\qquad w_3(k+n_k) - w_3(k)) \end{aligned} \tag{5.61}$$

Die Struktur für Gelenk 5 ist auch in Abb. 5.11 dargestellt[20]. Dabei sind die die absolute Position beschreibenden Variablen nicht einzeln aufgeschlüsselt.

Experimentell wird eine Struktur mit zwei verdeckten Schichten mit 7 und 3 bzw. mit 8 und 3 Neuronen ermittelt. Mit den Schwellwerten ergeben sich in der Ausgangsschicht 4 Gewichte, in der zweiten verdeckten Schicht $3 \cdot 8$ bzw. $3 \cdot 9$ und in der ersten verdeckten Schicht $7 \cdot 15$ bzw. $8 \cdot 25$ Gewichte. Insgesamt sind dies 133 bzw. 231 Gewichte.

Wegen des beschränkten Ausgangsbereichs von Neuronalen Netzen ist eine Normalisierung nötig, um das Netz an die Aufgabe anzupassen. So wird der Ausgangsbereich der Neuronen zur nichtlinearen Vorsteuerung auf das Intervall

[20] Zur besseren Vergleichbarkeit mit Abb. 5.4 werden die allgemeinen Variablen w und u durch q_d und q_c ersetzt. q_d ohne weiteren Index stellt den ganzen Vektor für den aktuellen Zeitpunkt dar.

(-3 mrad, 3 mrad) abgebildet. Üblicherweise wird auch der Eingang normalisiert, sodass die Netzeingänge auch etwa im Bereich (0, 1) liegen.

Während beim Training mit dem linearen Ansatz der Unterschied zwischen der Soll-Stellgröße und dem trainierten Wert nur durch Störungen und deren Filterung kommt, ist bei Neuronalen Netzen auch bei eindeutigen Funktionen nicht immer klar, wie das Netz die gewünschte Abbildung erfüllt. Aus diesem Grund wird ein Netzfehler

$$e_{Netz} = u_{neu} - u_{lin} - u_{Netz} \tag{5.62}$$

definiert, dessen quadratischer Mittelwert im Training minimiert werden soll. Dabei beschreibt u_{lin} die ersten Ausdrücke von Gleichung (5.60) und (5.61), während u_{Netz} die verbleibenden Anteile, also die Funktionen f_4 bzw. f_5 repräsentiert. Hier werden also die tatsächlichen physikalischen Größen verglichen, nicht die normierten Werte.

Das Training erfolgt dann durch das erweiterte Kalman Filter (Anhang B), das sich im Vergleich mit anderen Trainingsverfahren als geeignet herausgestellt hat. An dieser Stelle soll nicht weiter auf das Trainingsverfahren eingegangen werden. Einige vom Verfahren unabhängige Überlegungen sollen aber doch wiedergegeben werden.

So ist es zur schnellen Konvergenz hilfreich, wenn die Reihenfolge der Trainingsdaten verändert wird. Dies gilt insbesondere bei Verfahren mit schrittweiser Korrektur und bei Trainingsdaten, die zeitlich korreliert sind, wie z. B. bei Werten, die beim Abfahren einer Trajektorie entstehen. Durch das offline Training kann man nach Abfahren einer Trajektorie mit ungefähr N Schritten, also nach Aufnahme von N Datensätzen, die jeweils aus den Netzeingängen und dem Sollwert für den Ausgang bestehen, die Reihenfolge ändern. Bewährt hat sich der heuristische Ansatz, jeweils die $\sqrt{N}$ nächsten Datensätze zu überspringen und am Ende der Datei auf den ersten in dieser Epoche[21] untrainierten Datensatz zurück zu springen. Im Beispiel eines Trainings mit nur $N = 10$ Datensäzten wird also nach dem 1. Datensatz gleich der 4. trainiert, danach der 7. und der 10.. Danach wird der 2. Datensatz trainiert, gefolgt vom 5. und 8.. Schließlich folgen der 3., der 6. und der 9. Datensatz, bevor das Training wieder

[21] Das Training erfolgt iterativ, wobei das einmalige Abarbeiten aller Trainingsdaten als Epoche bezeichnet wird. Das Training besteht also aus mehreren Epochen.

von vorne beginnt.

Grundsätzlich wird nur aufgrund der Trainingsdaten der letzten Trajektorie trainiert. Aufgrund des iterativen Trainings unter Annahme einer Zeitvarianz (siehe Anhang B) wird dabei die Information aus den vorangegangenen Trajektorien abgeschwächt. Dies beschleunigt die Konvergenz.

Auf der anderen Seite ist es sinnvoll, das Training abzubrechen, sobald eine merkliche Reduktion des Netzfehlers erreicht wurde, da sonst die vorangegangen Trajektorien vollständig „vergessen“ werden. Ansonsten besteht die Gefahr, dass die Messstörungen einer Trajektorie großen Einfluss haben (overfitting). Bei den folgenden Experimenten wird deshalb das Training abgebrochen, sobald der Netzfehler halbiert ist, wobei dies allerdings nur am Ende jeder Epoche überprüft wird. Ein zweites Abbruchkriterium begrenzt die Zahl der Epochen auf 10. Es zeigt sich, dass der Netzfehler mit den Daten einer weiteren Trajektorie schneller abnimmt als bei Fortsetzung des Trainings aufgrund der alten Trainingsdaten, da man insgesamt kein rein zyklisches Lernen hat [25, 98]. Das dritte Abbruchkriterium beendet das Training, wenn ein Netzfehler von 0.030 mrad erreicht ist.

Schließlich kann noch bemerkt werden, dass die Rechenzeit neben den Einflussgrößen der linearen Schätzung, wie Zahl der Schätzwerte und der Trainingsdaten, auch maßgeblich von der Zahl an nötigen Epochen abhängt. So kann ggf. eine lange Trajektorie genauso schnell trainiert werden wie ein kurze, wenn aufgrund der größeren Zahl an Trainingsdaten zum Training weniger Epochen gebraucht werden.

5.5.4 Vereinfachte Berücksichtigung der Kopplungen der Handachsen

Bei den Handachsen gibt es eine spezielle Art von Kopplungen, die nicht von der Kinematik des Roboters bestimmt sind, sondern vom Antrieb. Dies gilt für Roboter, bei denen die Antriebe der Handachsen z. B. beim 3. Gelenk liegen. Bei so einem Aufbau bewegen sich bei Rotation des Motors des 4. Gelenks auch die Gelenke 5 und 6, da deren Antriebe durch das 4. Gelenk hindurchgeleitet werden.

Soll also nur das 4. Gelenk bewegt werden, so müssen sich die Motoren aller

drei Handachsen bewegen. Bei langsamen Bewegungen ist dies kein Problem, da diese Kopplung durch die Übersetzungsverhältnisse bestimmt und in der kinematischen Transformation des Roboters berücksichtigt ist. Bei schnellen Bewegungen zeigen sich Einflüsse.

Zur Anwendung des bislang beschriebenen Verfahrens muss man sich nun entscheiden, ob man eine Vorsteuerung für jede Achse ansetzt oder für jeden Motor. Die Vorsteuerung für jede Achse nimmt an, dass die Armträgheitsmomente, ggf. mit Lastmasse, dominant sind. Die Betrachtung der einzelnen Motoren geht dagegen davon aus, dass die drehenden Anteile der Motoren den größten Teil der Trägheit ausmachen. Ohne Last wird das der Fall sein. Aufgrund der relativ geringen Getriebeübersetzungen der Handachsen kann aber auch der andere Ansatz nicht verworfen werden. Beide Ansätze können kommen also infrage.

Der Ansatz einer unbekannten Nichtlinearität, die durch ein Neuronales Netz berücksichtigt wird, ist aber auch übertrieben. Hier handelt es sich zwar auch um Kopplungen zwischen den einzelnen Achsen. Es ist aber bekannt, dass diese Kopplungen im Gegensatz zu den in Abschnitt 5.5.3 beschriebenen linear sind. Man kann also anstelle eines Netzes eine lineare Vorsteuerung wie in Abschnitt 5.4.1 vorsehen. Dem nichtlinearen Charakter wird lediglich dadurch Rechnung getragen, dass die Eingangsgrößen von einer anderen Achse stammen.

Die erweiterte Gleichung (5.53) von Achse 6 lautet mit den Koppeltermen (siehe auch Abb. 5.12)

$$\begin{aligned} q_{c6}(k) = q_{d6}(k) \;+\; & r_{w61} \cdot (q_{d6}(k+n_t+1) - q_{d6}(k)) + \cdots \\ & \qquad + r_{w6n_w} \cdot (q_{d6}(k+n_t+n_w) - q_{d6}(k)) \\ +\; & r_{c61} \cdot (\tau^c_{d6}(k+n_t+1) - \tau^c_{d6}(k)) + \cdots \\ & \qquad + r_{c6n_c} \cdot (\tau^c_{d6}(k+n_t+n_c) - \tau^c_{d6}(k)) \\ +\; & r_{k641} \cdot (q_{d4}(k+n_t+1) - q_{d4}(k)) + \cdots \\ & \qquad + r_{k64n_{k64}} \cdot (q_{d4}(k+n_t+n_{k64}) - q_{d4}(k)) \\ +\; & r_{k651} \cdot (q_{d5}(k+n_t+1) - q_{d5}(k)) + \cdots \\ & \qquad + r_{k65n_{k65}} \cdot (q_{d5}(k+n_t+n_{k65}) - q_{d5}(k)). \end{aligned} \tag{5.63}$$

Gegenüber Gleichung (5.53), die für jede Achse gilt, ist die Notation hier insofern erweitert, dass die Achsen als zusätzliche Indizes angegeben sind. So ist bei

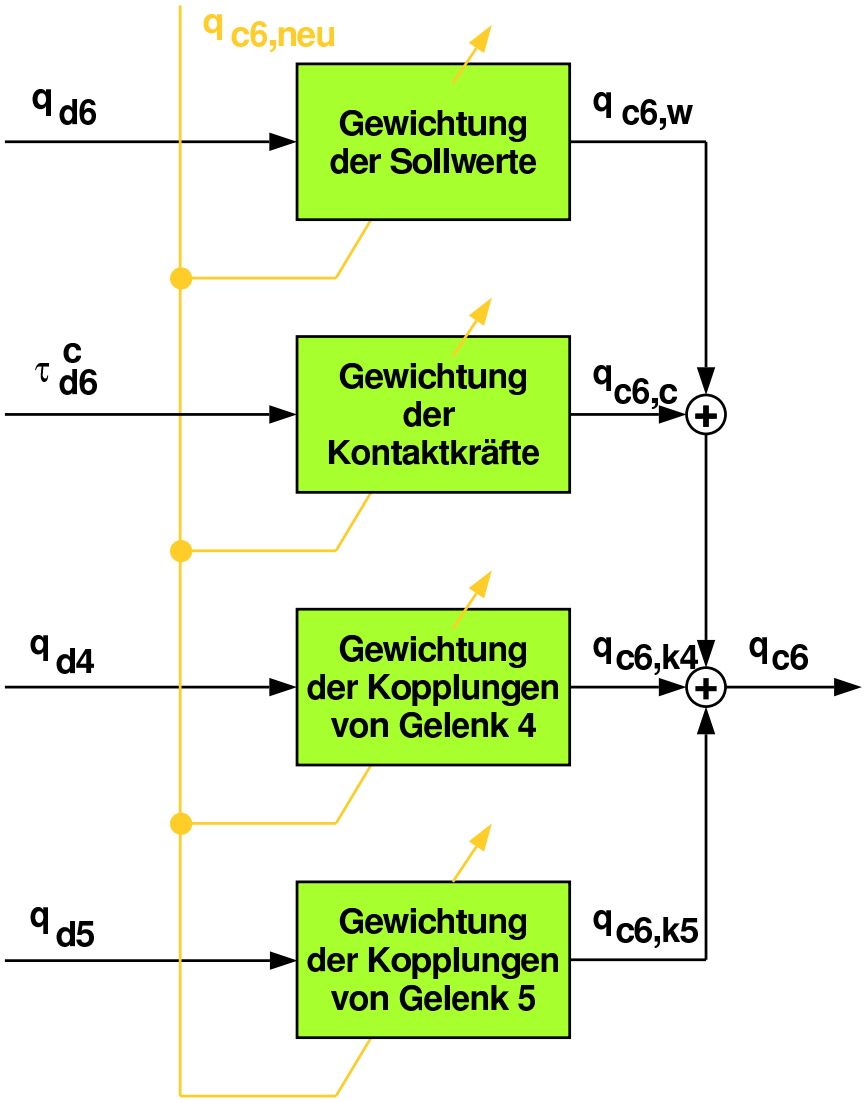


Abbildung 5.12: Struktur der zusätzlich zur internen Kaskadenregelung vorgeschlagenen Vorsteuerungen für Gelenk 6 (hell gezeichnet ist die Adaption aufgrund der optimierten Stellfolge; die Nullpunktverschiebungen sind nicht dargestellt)

Gelenkwinkeln und Reglerparametern ein zweiter Index eingeführt, der angibt, auf welche Achse sich die Größe bezieht. Bei den Reglerparametern $r_{k\ldots}$ wird zusätzlich angegeben, welche Achswerte gewichtet werden. r_{kijl} gewichtet also den Abtastwert $k+n_t+l$ von Achse j von q_d bei der Berechnung von $q_{ci}(k)$. $q_{ci,kj}$ ist die Summe der n_{kij} Koppelterme von Achse j auf die Bewegungsanweisung von Achse i.

Gleichung (5.63) entspricht u_{lin} in Gleichung (5.62). Die Ansätze für die anderen Handachsen sind entsprechend.

5.6 Adaption bei nichtlinearen Gütemaßen

Bisher wurde bei allen Identifikations, Optimierungs- oder Adaptionsaufgaben ein quadratisches Gütekriterium minimiert, da die numerische Lösung dadurch einfach ist. Außerdem ergeben quadratische Gütekriterien bei linearen Regelstrecken lineare Regler. Es gibt aber auch Anwendungen, bei denen ein anderes Gütemaß erforderlich ist. Dies erfordert theoretisch auch nichtlineare Regler. In der Praxis zeigt sich aber, dass die bisher vorgeschlagenen Regleransätze ausreichen. In den meisten Fällen bringt schon der lineare Regler bei Optimierung mit nichtquadratischem Gütekriterium eine deutliche Veränderung des Roboterverhaltens.

5.6.1 Explizite Berücksichtigung der Extremwerte

In Abschnitt 5.2 erfolgte die Schätzung der Änderungen der Kommandos durch Minimierung des quadratischen Fehlers von Gleichung (5.23) oder Gleichung (5.30) unter impliziter Berücksichtigung der Größe der Änderungen. Diese Berücksichtigung der Änderungen der Kommandos bewirkt z. B. bei rauhen Soll-Bahnen, dass auch bei idealer Modellkenntnis die gewünschte Trajektorie nicht genau erreicht wird. Stattdessen werden die Ecken der Soll-Bahn verschliffen. Der Versuch, die Bahn möglichst genau abzufahren, wird spätestens durch die maximal möglichen Motormomente begrenzt, die keine beliebig rauhe Bahn erlauben.

Wenn es keine sowohl praktikable als auch fehlerfreie Lösung von Gleichung (5.23) gibt, stellt sich die Frage, wie ein Kompromiss aussehen kann. Üblicherweise minimiert man den quadratischen Erwartungswert der Fehler (Gleichungen (5.24) bzw. (5.26)), da dies zu linearen Gleichungen bei der Optimierung führt. In der Praxis spielt dagegen der maximale Fehler eine Rolle [153]. Dieser Fehler wird angegeben, wenn ein Roboterhersteller die Genauigkeit angibt. Außerdem ist bei der Beurteilung eines bearbeiteten Werkstücks der maximale Fehler dafür entscheidend, ob das Werkstücks verwendbar ist oder ob es sich um Ausschuss handelt.

Die Minimierung eines nichtquadratischen Gütekriteriums erfordert prinzipiell einen anderen Ansatz als Least-Square Verfahren oder das verwendete Kalman Filter. Im Sonderfall der Minimierung der Extremwerte der Abweichungen

können die genannten Verfahren aber angewendet werden, allerdings modifiziert.

Die exakte Lösung der Minimierung des maximalen Fehlers betrachtet bei jedem Optimierungslauf nur die Zeile von Gleichung (5.23), in der die maximale Regeldifferenz $w(n_t+i)-y(n_t+i)$ auftritt. Aufgrund dieser einen Zeile werden die Stellgrößen der einzelnen Abtastschritte korrigiert. Dieser Ansatz erfordert zur Adaption aber sehr viele Trainingsläufe. Daher wurde er als unpraktikabel verworfen.

Eine Näherung der exakten Lösung kann mit dem Kalman Filter Ansatz erreicht werden, indem die angenommene Varianz der Messstörung σ^2 zeitabhängig gewählt wird. Bei großen Regeldifferenzen wird die tatsächlich herrschende Störung angenommen, bei kleinen Abweichungen wird die Störungsannahme dagegen stark erhöht. Das führt dazu, dass kleine Regeldifferenzen kaum berücksichtigt werden.

Der Ansatz

$$\sigma(k) = \begin{cases} \sigma_0 & \text{für} \quad e_{max} \leq |e(k)| \\ \sigma_0 \cdot (e_{max}/|e(k)|)^\alpha & \text{für} \quad e_{min} < |e(k)| < e_{max} \\ \sigma_0 \cdot (e_{max}/e_{min})^\alpha & \text{für} \quad |e(k)| \leq e_{min} \end{cases} \tag{5.64}$$

führt zu einer stetigen Gütefunktion anstelle des Sprungs bei $e(k) = e_{max}$. e_{max} und e_{min} sind dabei positiv angenommen. Die dritte Zeile ist aus numerischen Gründen nötig, um σ nach oben zu begrenzen. $\alpha = 8$ ist ein sinnvoller Wert für die Gewichtung der Größe der Regeldifferenz.

Die nichtlineare Anpassung von σ^2 an die Regeldifferenz wird für jede Achse individuell vorgenommen.

Die nichtlineare Wahl von $\sigma^2(k)$ in Abhängigkeit der Regeldifferenz $e(k)$ reduziert zwar den maximalen Fehler in Konfliktfällen, sie ist aber auch nicht geeignet, um in die Nähe der optimalen Trajektorie zu kommen. Bei ausschließlicher Anwendung von Gleichung (5.64) konvergiert das System immer noch sehr langsam.

Deshalb wird vorgeschlagen, zuerst mit festem σ^2 zu optimieren, bis der minimale quadratische Fehler erreicht ist. In weiteren Iterationen kann dann auf Gleichung (5.64) umgeschaltet werden. Dadurch wird der maximalen Fehler zu-

lasten des mittleren Fehlers verbessert. Schließlich ergibt sich die Lösung mit minimalem maximalen Fehler, jedoch mit leicht erhöhtem mittleren Fehler.

5.6.2 Kartesische Gütemaße

Eine andere Art von Kompromiss für die Minimierung von Gleichung (5.23) geht davon aus, dass in der Praxis nur der kartesische Positionsbahnfehler und ggf. der kartesische Orientierungsbahnfehler von Bedeutung sind, nicht aber die Positionsfehler auf Achsebene [153]. Die Experimente zeigen, dass zur Reduktion des kartesischen Bahnfehlers eine zahlenmäßig wesentlich größere Verbesserung der Achsfehler erforderlich ist, da die Achsfehler sich unter Umständen gegenseitig kompensieren. Die Reduktion einzelner Achsfehler kann dadurch sogar zu einer Zunahme des kartesischen Fehlers führen.

Es kann auch vorkommen, dass die Reduktion *aller* Achsfehler zur Verbesserung kartesischer Komponenten führt, die für die entsprechende Anwendung keine oder nur geringe Bedeutung haben, z. B. die Orientierung des Endeffektors. Die Beschränkung der Orientierungsgenauigkeit könnte aber zu einer Erhöhung der Genauigkeit der translatorischen Komponenten führen.

Auf der anderen Seite gibt es Fälle, in denen die Regeldifferenzen zunächst kartesisch vorliegen [154], z. B. bei externer Vermessung der Ist-Werte des Roboters (vergl. Abschnitt 4.3.4). Auch dabei macht es wenig Sinn, die kartesischen Werte zuerst in Gelenkwinkel umzurechnen, was meist nicht eindeutig ist.

Daher wird vorgeschlagen, Gleichung (5.23) in dem Raum zu optimieren, in dem das Gütekriterium sich geeignet ausdrücken lässt [153]. Dies erfordert zunächst eine gemeinsame Betrachtung aller Achsen und dann noch die Transformation der Regeldifferenzen, Modelldaten und Störungsannahmen durch die Jacobi-Matrix $\mathbf{J}$. Die Betrachtung aller Achsen ersetzt die Skalare Δu und e durch Vektoren, die als Elemente die jeweiligen Gelenkwerte enthalten. Das geschätzte i-te Element der Gewichtsfunktion (Impulsantwort) $\hat{g}_i$ wird durch die Diagonalmatrix $\hat{G}_i$ ersetzt, die als Elemente die Werte $\hat{g}_i$ der einzelnen Gelenke enthält. Mit diesen Bezeichnungen ergibt sich aus Gleichung (5.23)

$$\begin{pmatrix} \mathbf{\Phi J}(1) & & \\ & \ddots & \\ & & \mathbf{\Phi J}(N) \end{pmatrix} \cdot \begin{pmatrix} \hat{\mathbf{G}}_1 & & \\ \vdots & \ddots & \\ \hat{\mathbf{G}}_{n_g} & & \ddots \\ & \hat{\mathbf{G}}_{n_g} \cdots \hat{\mathbf{G}}_1 & \end{pmatrix} \cdot \begin{pmatrix} \Delta\mathbf{u}(0) \\ \vdots \\ \vdots \\ \Delta\mathbf{u}(N-1) \end{pmatrix} = \begin{pmatrix} \mathbf{e}_\Phi(n_t+1) \\ \vdots \\ \vdots \\ \mathbf{e}_\Phi(N+n_t) \end{pmatrix} \tag{5.65}$$

Dabei ist noch die Abkürzung

$$\mathbf{e}_\Phi = \mathbf{\Phi} \cdot \mathbf{J} \cdot \mathbf{e} \tag{5.66}$$

verwendet. $\mathbf{\Phi}$ ist eine Selektionsmatrix, die diejenigen Elemente der kartesischen Darstellung auswählt, die minimiert werden sollen. Durch Rechteckform der $(p \times 6)$ Matrix $\mathbf{\Phi}$ reduziert sich die Gesamtzahl der Gleichungen auf $p \cdot N$ Gleichungen bei p interessierenden Komponenten des kartesischen Vektors.

Die Wahl des kartesischen Systems ist grundlegend für $\mathbf{\Phi}$. Normalerweise soll der Bahnfehler minimiert werden, also der Positionsfehler senkrecht zur Bewegungsrichtung, und nicht der Schleppfehler, also der Positionsfehler in Bewegunsgrichtung. Man erhält eine konstante Selektionsmatrix $\mathbf{\Phi}$, wenn man das kartesische System zeitvariant so wählt, dass die Bewegungsgrichtung in jedem Schritt z. B. der z-Richtung entspricht. Dann wählt die Selektionsmarix beispielsweise die x- und die y-Komponte aus, wenn man den Bahnfehler minimieren will. Ein ähnliches zeitvariantes Koordinatensystem wurde auch schon in den Abschnitten 3.4 und 4.3.4 ausgewählt.

Wenn das gewählte kartesische Koordinatensystem nicht das Basissystem des Roboters ist, muss man die Berechnung der Jacobi-Matrix $\mathbf{J}$ entsprechend modifizieren. Dann beschreibt $\mathbf{J}$ die partiellen Ableitungen der kartesischen Werte nach den Achswerten in dem gewählten System.

Die Schätzwerte von Gleichung (5.65) sind gegenüber Gleichung (5.23) unverändert Achswerte. Deshalb ist die Adaption der Vorsteuerung oder einer Messwertrückführung von der kartesischen Minimierung nicht betroffen.

Zu erwähnen ist noch, daß die Störungsannahme in gleicher Weise wie die Regeldifferenz nach Gleichung (5.66) transformiert werden muss.

$$\sigma_\Phi = \mathbf{\Phi} \cdot \mathbf{J} \cdot \sigma \tag{5.67}$$

Hierbei sind σ_Φ und σ Vektoren mit den Störungsannahmen in den kartesischen Richtungen bzw. in den einzelnen Achsen.

Eine Kombination der Methoden dieses Abschnitts und des vorigen Abschnitts, also die Minimierung des maximalen kartesischen Bahnfehlers, ist möglich. In dem Fall erfolgt die Anpassung der Störungsannahmen kartesisch, wobei am besten σ_0 entsprechend Gleichung (5.67) angepasst wird.

Die Minimierung eines Fehlers mit weniger Freiheitsgraden als der Zahl der Gelenke bedeutet, dass die übrigen Freiheitsgrade nicht betrachtet werden. In diesen Komponenten ist also ein beliebiger Fehler möglich. Zum anderen ist die Trennung zwischen Bahn- und Schleppfehler bei gekrümmten Bahnen mit großem Schleppfehler nicht eindeutig. Ein sehr großer Schleppfehler ist also unter Umständen auch für den Bahnfehler nachteilig.

Daher wird vorgeschlagen, zunächst die einzelnen Achsfehler zu reduzieren. Wenn die Achsfehler sich nicht weiter verbessern lassen, kann der kartesische Bahnfehler durch Anwendung von Gleichung (5.65) weiter reduziert werden. Danach ist ggf. eine spezielle Berücksichtingung des maximalen kartesischen Bahnfehlers möglich.

Dabei werden die jeweils nicht berücksichtigten Komponenten oder Werte sich leicht verschlechtern, um die Verbesserung der im Gütekriterien betrachteten Komponenten zu ermöglichen. Unter Umständen kann es auch zu einer Drift von nicht berücksichtigten Anteilen kommen. In dem Fall muss man das Training rechtzeitig beenden.

5.7 Resümee

Das Kapitel zeigt die dreistufige Adaption des Reglers. Dabei besteht die Regelstrecke nicht nur aus dem Industrieroboter selbst sondern aus dem durch die industriell eingesetzte Steuerung geregelten System.

In der ersten Adaptionsstufe wird ein Modell identifiziert. Diese Adaption erfolgt einmalig an einer Beispieltrajektorie. Die Modellgüte ist im Gegensatz

Einfluss	Bezeichnung des Reglerelements
-	lineare Positionsvorsteuerung (Gewichtung zukünftiger Soll-Positionen des betrachteten Gelenks)
Kopplungen zwischen den Roboterarmen	nichtlineare Positionsvorsteuerung durch Neuronale Netze (Gewichtung aktueller Werte aller Gelenkwinkel und zukünftiger Soll-Positionen der beeinflussenden Gelenke)
antriebsbedingte Kopplungen der Handachsen	vereinfachte Entkopplung der Handachsen (Gewichtung zukünftiger Soll-Positionen der beeinflussenden Gelenke)
Kontaktkraft	Kraftvorsteuerung (Kontaktkraftkompensation) (Gewichtung zukünftiger Kontaktkräfte)
schlechte Reproduzierbarkeit der industriellen Steuerung	Positionsrückführung (Gewichtung vergangener Positionsregelfehler und Bewegungskommandos)
schlechte Reproduzierbarkeit der industriellen Steuerung und Kraftkontakt	Kraftrückführung (Gewichtung vergangener Kontaktkräfte)

Tabelle 5.2: Elemente des vorgeschlagenen Reglers

zu modellbasierten Regelungsansätzen von untergeordeneter Bedeutung, da die weiteren Stufen der Adaption durch inkrementelle Verbesserungen leichte Modellfehler ausgleichen.

Die beiden anderen Adaptionsstufen werden mehrfach unmittelbar hintereinander aufgerufen. Dabei optimiert die zweite Stufe im Nachhinein die für die betrachtete Trajektorie erforderlichen Stellgrößen. Die dritte Stufe führt eine Parameteradaption durch, um den Regler so anzupassen, dass er die berechneten Stellgrößen aus den beim Abfahren der Trajektorie verfügbaren Soll- und Messwerten generiert.

In Abhängigkeit der Aufgabenstellung und der Annahmen über existierende Störungen sind unterschiedliche Betriebsarten und Regleransätze vorteilhaft. Bei immer wiederkehrenden festen Trajektorien ist kein Regler nötig. In dem Fall kann die optimierte Stellfolge direkt kommandiert werden. Andernfalls sind ein oder mehrere Reglerelemente[22] nach Tabelle 5.2 erforderlich.

All diese Reglerelemente werden prinzipiell durch Parameterschätzverfahren adaptiert. Bei den linearen Elementen ist dies eine spezielle Interpretation des Kalman Algorithmus, bei Neuronalen Netzen werden die Gewichte durch EKFNet, ein auf dem erweiterten Kalman Filter basierenden Trainingsverfahren so be-

[22]Ein Regler, der keine vergangenen Werte berücksichtigt, wird in dieser Arbeit auch als Vorsteuerung oder Vorfilter bezeichnet

stimmt, dass die durch die zweite Adaptionsstufe vorgegebenen Stellgrößen generiert werden.

Die zur Steuerung der einzelnen Adaptionen vorzugebenden Parameter der Adaptionsverfahren lassen sich spätestens nach Abschluss der Identifikation leicht vorgeben. Auch kann die Qualität der Adaption aus den Ergebnissen abgeschäzt werden. Die Vorgaben der Parameter beruhen darauf, dass die für eine Stufe vorzugebenden Varianzen sich aus der tatsächlichen oder geschätzten Güte der vorangegangenen Adaptionsstufen herleiten lassen. Eine ausführliche Diskussion dieses Themas kann in dieser Arbeit aus Platzgründen nicht erfolgen. Festzuhalten bleibt, dass abgesehen von der unkritischen Identifkation vom Benutzer keine Steuerungsparameter für die Adaption vorzugeben sind. Damit sind Stabilität und Konvergenz automatisch gewährleistet.

Kapitel 6

Experimentelle Bewertung der adaptiv vorausplanenden Steuerung

In diesem Kapitel wird die adaptiv vorausplanende Steuerung experimentell untersucht. Dabei erfolgen die Experimente an unterschiedlichen Industrierobotern und mit unterschiedlichen Aufgabenstellungen. Die durch die einzelnen Komponenten des vorgeschlagenen Verfahrens erreichbaren Bahnverbesserungen werden qualitativ und quantitativ bewertet.

6.1 Verhalten bei rein linearen Vorsteuerungen

6.1.1 Training der Vorsteuerung

Zur Bestimmung der Parameter der Vorsteuerung wird zunächst eine Testbahn aus Anhang C langsam, aber mit Anregung abgefahren, um die Gewichtsfunktion zu bestimmen. Danach wird die Bahn[1] mit hoher Geschwindigkeit und ohne Anregung durchlaufen. Das ist die Ausgangsbahn, deren Bahngenauigkeit dann verbessert wird. Sie wird in den folgenden Tabellen als 0. Iteration bezeichnet und entspricht im Rahmen der Wiederholgenauigkeit Tabelle 3.2.

Zur Darstellung der Verbesserung ist es nicht sinnvoll, Soll-Bahnen und gemessene Trajektorien ähnlich wie in Abb. 3.14 in Diagrammen zu zeigen. Sie stimmen nach wenigen Iterationen im Rahmen der Zeichengenauigkeit überein.

[1] Prinzipiell kann bei jedem Testlauf eine neue Trajektorie gewählt werden. Zur Verfolgung der Verbesserung ist es aber vorteilhaft, immer die gleiche Testbahn zu verwenden, oder, zum Training einer nahezu ortsunabhängigen Vorsteuerung, eine kleine Auswahl von grundverschiedenen Bahnen.

	Manutec r2	Kuka KR6/1
0	39.893 mm	59.604 mm
1	6.024 mm	17.861 mm
2	1.205 mm	5.739 mm
3	0.441 mm	1.804 mm
4	0.310 mm	0.637 mm
5	0.253 mm	0.381 mm
6	0.232 mm	0.348 mm
7	0.225 mm	0.339 mm
8	0.219 mm	0.335 mm
9	0.214 mm	0.321 mm
10	0.211 mm	0.318 mm

Tabelle 6.1: Abnahme der mittleren kartesischen Positionsfehler während des Trainings der Vorsteuerung

Auch lassen sich die Positionsfehler nur über wenige Iterationen in einem Diagramm über der Zeit darstellen (siehe Abb. 6.1), da sie sich im Laufe der Adaptionen um Größenordnungen ändern. Deshalb bleibt nur die Auflistung der mittleren Positions- und Bahnfehler der einzelnen Iterationen. Diese Mittelwerte lassen sich bei halblogarithmischer Achseneinteilung auch grafisch darstellen.

Tabelle 6.1 (grafische Darstellung in Abb. 6.11) zeigt das Training an Bahn 3, die aus horizontalen und vertikalen Kreissegmenten besteht (Darstellung der Bahnen in Anhang C). Nach einmaligem Abfahren mit 250 mm/s zur Modellbildung (nicht aufgeführt) wird die Bahn mehrfach mit der Soll-Geschwindigkeit von 500 mm/s durchlaufen. Dabei nehmen die Positionsfehler innerhalb weniger Iterationen stark ab. Bei der hohen Geschwindigkeit lässt sich der mittlere kartesische Positionsfehler nach Gleichung (3.3) innerhalb der 10 gezeigten Iterationen fast um den Faktor 200 bis auf wenige zehntel Millimeter reduzieren. Dabei erfolgen die wesentlichen Änderungen schon innerhalb der ersten beiden Iterationen.

Bei Wiederholung des Trainings unter gleichen Bedingungen ergeben sich Unterschiede im erreichbaren Positionfehler von bis zu 10 %. Die Wiederholung einer Bahn mit identischer Vorsteuerung ist dagegen im Rahmen der Wiederholgenauigkeit der Roboter identisch. Die Adaption der Vorsteuerung reagiert also empfindlicher als die Regelung der vorgesteuerten Bahn.

Die Tabellen 6.2 und 6.3 vergleichen die einzelne Gütemaße vor und nach einem durchschnittlichen Training. Die Verbesserung des Bahnfehlers, also der Komponente des Positionsfehlers, die senkrecht zur Bahn steht, ist deutlich geringer

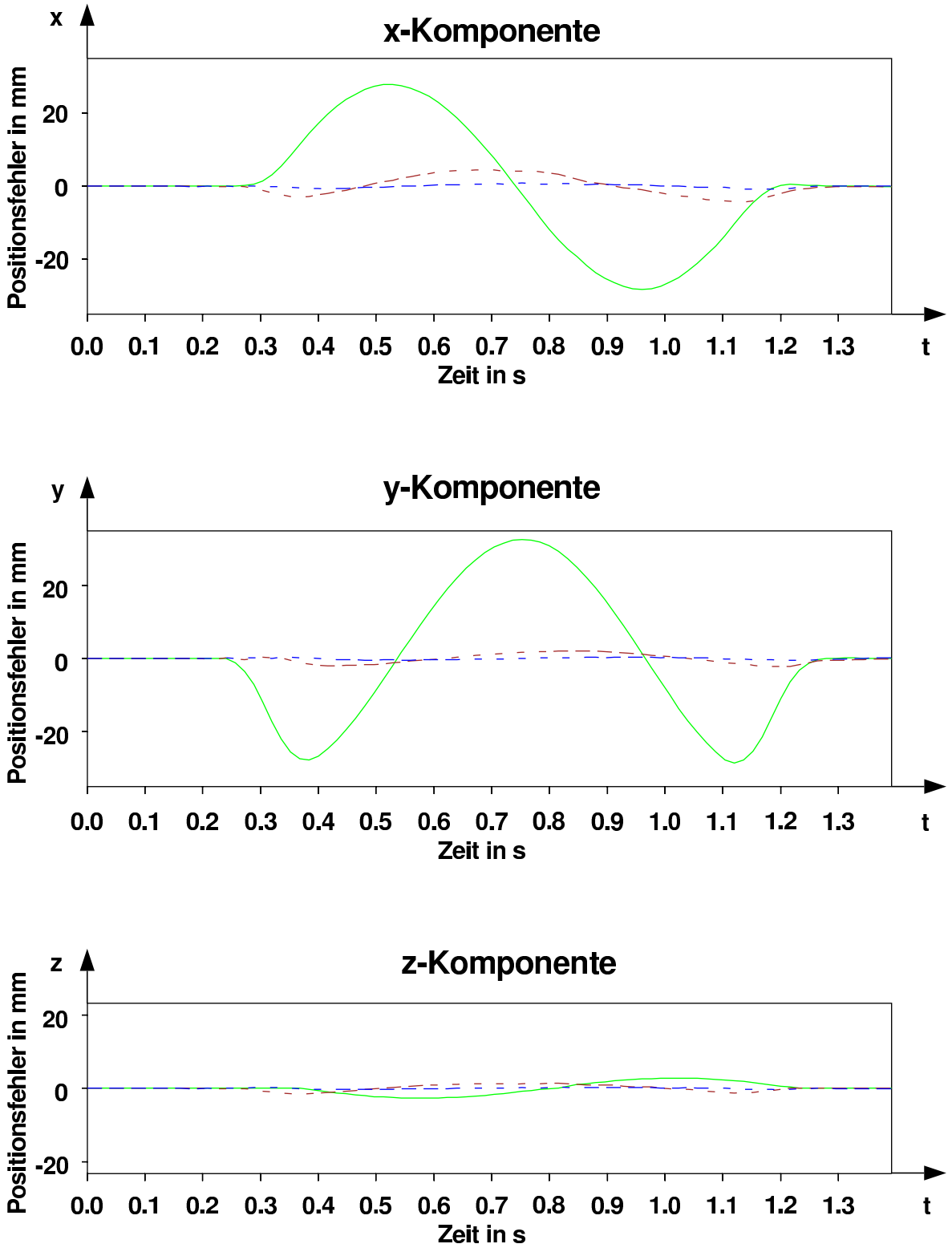


Abbildung 6.1: Positionsfehler des Manutec r2 Roboters beim Abfahren von Bahn 1 mit 375 mm/s vor der Adaption (grün durchgezogen), nach der ersten (braun gestrichelt) und der zweiten (blau strichpunktiert) Iteration

	ohne Vorsteuerung	mit Vorsteuerung
Positionsfehler von Achse 1	82.816 mrad	0.313 mrad
Positionsfehler von Achse 2	61.493 mrad	0.185 mrad
Positionsfehler von Achse 3	97.277 mrad	0.311 mrad
Positionsfehler von Achse 4	0.260 mrad	0.499 mrad
Positionsfehler von Achse 5	70.682 mrad	0.457 mrad
Positionsfehler von Achse 6	92.699 mrad	1.040 mrad
kartesischer Positionsfehler	39.884 mm	0.211 mm
kartesischer Bahnfehler	6.009 mm	0.158 mm

Tabelle 6.2: Verbesserung der Genauigkeit des Manutec r2 Roboters durch Vorsteuerung von Bahn 3

als die Verbesserung des Positionsfehlers selbst. Das kommt daher, dass die Reduktion der Positionsfehler hauptsächlich den vorher dominanten Schleppfehler verringert. Das zeigt die Problematik, dass die Vorsteuerung, die auf die einzelnen Achsen wirkt, zur Reduktion des Bahnfehlers eine wesentlich höhere Reduktion der einzelnen Achsfehler leisten muss.

Die verbleibenden Fehler kommen durch Nichtlinearitäten, die im linearen Ansatz der Vorsteuerung nach Gleichung (5.44) nicht dargestellt werden können. Die hohe Genauigkeit zeigt, dass beim Manutec r2 Roboter die Trägheit der Armelemente gegenüber den Rotorträgheitsmomenten eine untergeordnete Rolle spielt, sodass schon mit dem linearen Ansatz eine so große Verbesserung der Positionsgenauigkeit möglich ist.

Auffällig ist der Fehler von Achse 4 beim Manutec r2 Roboter. Durch die Vorsteuerung wird dieser Achsfehler etwa verdoppelt. Abb. 6.2 zeigt den Positionsfehler mit Vorsteuerung über der Zeit. Diese Achse ist an der Soll-Bewegung nicht beteiligt, hat aber zu manchen Zeitpunkten trotzdem Positionsfehler, die deutlich über der Auflösung der Messwerte liegen. Diese Positionsfehler sind

	ohne Vorsteuerung	mit Vorsteuerung
Positionsfehler von Achse 1	70.577 mrad	0.255 mrad
Positionsfehler von Achse 2	54.054 mrad	0.197 mrad
Positionsfehler von Achse 3	60.721 mrad	0.250 mrad
Positionsfehler von Achse 4	0.053 mrad	0.049 mrad
Positionsfehler von Achse 5	51.408 mrad	0.517 mrad
Positionsfehler von Achse 6	71.746 mrad	0.907 mrad
kartesischer Positionsfehler	59.619 mm	0.318 mm
kartesischer Bahnfehler	1.825 mm	0.262 mm

Tabelle 6.3: Verbesserung der Genauigkeit des Kuka KR6/1 Roboters durch Vorsteuerung von Bahn 3

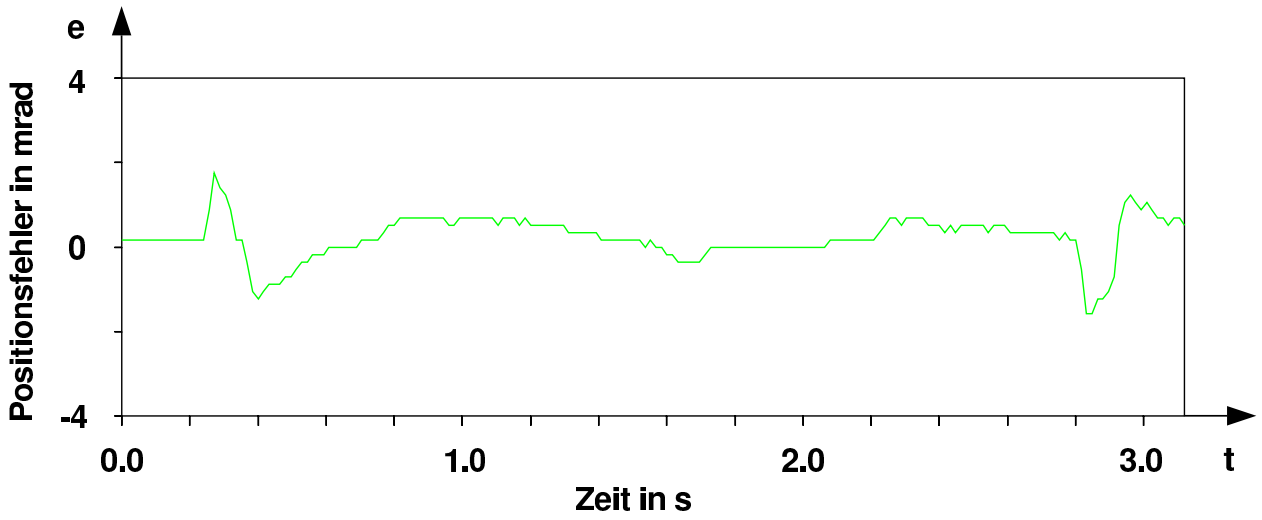


Abbildung 6.2: Positionsfehler von Achse 4 des Manutec r2 Roboters bei linearer Vorsteuerung aller Achsen

Kopplungen, die durch die Beschleunigung und das Abbremsen von Achse 1 am Anfang und Ende der Bahn (siehe auch Kinematik in Abbildung 3.13) verursacht werden. Durch die unterlagerte Regelung von Achse 4 führt die annähernd konstante Beschleunigung von Achse 1 zu einem stark ansteigenden Fehler, dann zu einem langsamen Ausgleich, der die Soll-Position übertrifft, da der Beschleunigungsvorgang beendet ist, und schließlich zum Einregeln auf die Soll-Position. Beim Abbremsen ist der umgekehrte Effekt zu beobachten.

Da diese Achse einen konstanten Sollwert hat, ist keine Verbesserung durch eine Vorsteuerung nach Abschnitt 5.4.1 möglich. Zur Verringerung des Positionsfehlers von Achse 4 ist also entweder eine Bahnsteuerung oder eine achsübergreifende Vorsteuerung nötig. Dies wird in den Abschnitten 6.4 und 6.2 ausgeführt.

Die Kopplungen haben beim Manutec r2 Roboter bei Achse 4 eine relativ starke Auswirkung, da das Gelenk nicht für so lange Werkzeuge wie den montierten Greifer (siehe Abb. 3.1) ausgelegt ist. Bei den anderen Achsen können die Positionsfehler nicht so leicht auf Kopplungen zurückgeführt werden, ein Einfluss ist aber sicher vorhanden.

6.1.2 Messung der Bahngenauigkeit an untrainierten Bahnen

Abschnitt 6.1.1 zeigt die Verbesserung der Ausführungsgenauigkeit von Bahn 3 durch Training an dieser Bahn. Dieses Training bewirkt auch bei anderen, untrainierten Bahnen oder Geschwindigkeiten eine Verbesserung. Dies ist in Tabelle 6.4 sowie den vorletzten Spalten der Tabellen 6.5 und 6.6 dargestellt.

Dabei sind die untersuchten Bahnen teilweise grundverschieden zur Trainingsbahn 3 (vergl. Anhang C). So ist die Orientierung des Greifers variiert, manche Bahnen beschreiben Rechtecke, andere Kreise, und der Arbeitsbereich unterscheidet sich stark von der Trainingsbahn.

Bei der Wahl der Bahnen wurde darauf aber geachtet, dass auch die veränderte Stellfolge die engen Beschleunigungsgrenzen der Robotersteuerung einhält. Es zeigt sich, dass die ansonsten eintretende Skalierung schon bei geringfügigen Überschreitungen in wenigen Abtastschritten den mittleren Positionsfehler der gesamten Bahn deutlich verschlechtert.

Die Verbesserung des Positionsfehlers liegt durchweg, also nicht nur bei der Trainingsbahn, über einem Faktor von 100, wobei langsame, also genauere Bewegungen sich naturgemäß weniger verbessern lassen. Die Bahntreue wird, abgesehen von den ohnehin genauen langsamen Bewegungen des Kuka KR6/1 Roboters, mindestens um den Faktor 10 verbessert.

Beim Manutec r2 Roboter wird der größte Fehler nach erfolgtem Training mit Bahn 3 bei Bahn 1 festgestellt, also bei einer relativ einfachen Kreisbahn. Dies legt die Vermutung nahe, dass sich das Roboterverhalten zwischen dem hängenden und dem stehenden Greifer stark unterscheidet.

Deshalb wird das Training wiederholt, diesmal aber unter Verwendung beider Bahnen. Dazu wird zuerst jede Bahn langsam abgefahren, um das Modell zu

		kartesischer Positionsfehler		kartesischer Bahnfehler	
Bahn	Geschw.	ohne Vorst.	mit Vorst.	ohne Vorst.	mit Vorst.
3	583 mm/s	59.619 mm	0.304 mm	1.825 mm	0.251 mm
3	250 mm/s	28.525 mm	0.183 mm	0.395 mm	0.120 mm
-	1250 mm/s	89.939 mm	0.267 mm	0.923 mm	0.099 mm
-	250 mm/s	26.568 mm	0.136 mm	0.103 mm	0.088 mm

Tabelle 6.4: Wirkung der an Bahn 3 mit 583 m/s für den Kuka KR6/1 Roboter trainierten Vorsteuerung auf die gerade Nominalbahn aus Abschnitt 3.3.2

Bahn	Geschw.	Positionsfehler ohne Vorsteuerung	Positionsfehler mit Vorsteuerung durch Bahn 3	Positionsfehler mit Vorsteuerung durch Bahn 1 und 3
1	375 mm/s	23.091 mm	0.245 mm	0.216 mm
1	188 mm/s	13.546 mm	0.174 mm	0.134 mm
2	375 mm/s	24.053 mm	0.210 mm	0.182 mm
2	188 mm/s	14.126 mm	0.161 mm	0.121 mm
3	500 mm/s	39.900 mm	0.209 mm	0.187 mm
3	250 mm/s	21.523 mm	0.157 mm	0.130 mm
4	375 mm/s	32.074 mm	0.183 mm	0.160 mm
4	188 mm/s	16.709 mm	0.156 mm	0.123 mm
5	375 mm/s	27.221 mm	0.206 mm	0.185 mm
5	188 mm/s	16.218 mm	0.180 mm	0.155 mm

Tabelle 6.5: Wirkung der trainierten Vorsteuerung auf den kartesischen Positionsfehler des Manutec r2 Roboters bei verschiedenen Bahnen (Bahnen siehe Anhang)

identifizieren. Danach werden die beiden Bahnen abwechselnd durchlaufen und die Daten zur Schätzung des Vorfilters verwendet.

Die letzten Spalten in den Tabellen 6.5 und 6.6 demonstrieren das Ergebnis nach 10 Iterationen. Für die beiden letzten Spalten wurde, abgesehen von dem 2. Identifikationslauf, gleich lang trainiert.

Interessanterweise sind die Ergebnisse allgemein besser, wenn die Vorsteuerung aufgrund von zwei grundverschiedenen Bahnen trainiert wird. Dadurch ist zwar bei allen vorher untrainierten Bahnen eine Verbesserung durch das größere Trainingsspektrum zu erwarten, die Tatsache, dass auch Bahn 3 durch das Training von Bahn 1 verbessert wird, muss aber als Zufall gelten.

Insgesamt wird durch die Tests bestätigt, dass die Vorsteuerung nicht ortsabhängig zu sein braucht. Es ist damit auch nicht zu erwarten, dass ein ortsabhängiges Modell mit ortsabhängiger Vorsteuerung zu einem geringeren Po-

Bahn	Geschw.	Bahnfehler ohne Vorsteuerung	Bahnfehler mit Vorsteuerung durch Bahn 3	Bahnfehler mit Vorsteuerung durch Bahn 1 und 3
1	375 mm/s	2.107 mm	0.125 mm	0.114 mm
1	188 mm/s	1.209 mm	0.093 mm	0.074 mm
2	375 mm/s	3.825 mm	0.125 mm	0.114 mm
2	188 mm/s	2.214 mm	0.101 mm	0.078 mm

Tabelle 6.6: Wirkung der trainierten Vorsteuerung auf den kartesischen Bahnfehler des Manutec r2 Roboters bei anderen Bahnen (Bahnen siehe Anhang)

sitionsfehler führt. Damit erweist sich sowohl die Vorsteuerung als auch die Adaption der Vorsteuerung als robust gegenüber Parameterschwankungen.

6.1.3 Verifikation der Messung

Die Verbesserungen der Bahngenauigkeit in den Abschnitten 6.1.1 und 6.1.2 beziehen sich auf die von der Robotersteuerung ausgegebenen Ist-Werte. In Abschnitt 3.4 wurde aber festgestellt, dass die tatsächliche (abtriebsseitige) Bahn sich von der antriebsseitig gemessenen unterscheidet. Dies soll in diesem Abschnitt weiter untersucht werden.

Als Testbahn wird, wie in Abschnitt 3.4, die Schmid'sche Bahn beim Kuka KR6/1 Roboter verwendet. Für diese Bahn wurde schon in Tabelle 3.6 festgestellt, dass die antriebsseitig trainierte Vorsteuerung real keine wesentliche Verbesserung bringt. Deshalb wird hier gezeigt, wie die Verwendung des externen Messsystems die Adaption der Vorsteuerung modifiziert. Zum Vergleich werden die Experimente mit dem Kuka KR15/2 wiederholt, einem Roboter mit anderen Getrieben, wodurch theoretisch mehr Nachgiebigkeit, aber weniger Lose auftritt.

Es werden folgende Schritte unternommen:

1. Herkömmliche Adaption einer Vorsteuerung aufgrund der internen Messwerte beim Abfahren von Bahn 3a.
2. Langsames Abfahren der Schmid'schen Bahn mit der Vorsteuerung und Aufzeichnen der Unterschiede zwischen interner und externer Messung zur Kalibrierung des externen Messsystems.
3. Schnelles Abfahren der Schmid'schen Bahn mit der Vorsteuerung und Aufzeichnen der Unterschiede zwischen intern und kalibriert extern gemessener Bahn. Dies sind die dynamischen Bahnabweichungen.
4. Iteratives Ausführen von
   (a) Schnelles Abfahren der Schmid'schen Bahn mit der Vorsteuerung.
   (b) Korrektur der intern gemessenen Position und Auswertung
   (c) Adaption der Vorsteuerung

bis der mittlere Bahnfehler sich nicht mehr verbessert.

5. Schnelles Abfahren der Schmid'schen Bahn und Auswertung der kalibriert extern gemessenen Bahn.

Der unter Punkt 5 gemessene Bahnfehler ist der tatsächliche Bahnfehler mit Vorsteuerung. Wenn er noch zu hoch ist oder sich deutlich von dem unter Punkt 4.b gemessenen unterscheidet, müssen die Punkte 3 bis 5 wiederholt werden, wobei das Abfahren der Trajektorie in Punkt 3 nicht wiederholt werden muss. Bei den folgenden Experimenten ist diese äußere Iteration nur beim Kuka KR15/2 Roboter nötig (vergl. Tabelle 6.8). Beim Kuka KR6/1 Roboter sind die Ergebnisse gleich ausreichend.

In Abschnitt 3.4 wurde schon aufgezeigt, dass sich der Einfluss der Nachgiebigkeit durch die Vorsteuerung ändert. Deshalb wäre die äußere Iteration in jedem Fall nötig, wenn man Punkt 1 weglassen würde.

Tabelle 6.7 fasst die Experimente mit dem Kuka KR6/1 Roboter zusammen. Die allgemeine Vorsteuerung nach Punkt 1 reduziert die motorseitg gemessenen Bahnabweichungen bei beiden Bahnen. Armseitig ändert sich der Verlauf auch, die Mittel- und Maximalwerte werden aber kaum reduziert (siehe Abb. 3.29 und 3.30).

Bahnfehler bei Messung durch	Encoder	Dynalog
Ausführung der Schmid'schen Bahn		
Ohne Vorsteuerung	0.766 (5.450)	0.626 (5.075)
Allgemeine Vorsteuerung	0.200 (1.269)	0.459 (3.225)
Mit ext. Messung adaptierte Vorsteuerung	0.312 (1.725)	0.202 (0.853)
Ausführung von Bahn 3a		
Ohne Vorsteuerung	1.834 (2.431)	1.325 (2.200)
Allgemeine Vorsteuerung	0.186 (0.508)	0.642 (1.531)
Mit ext. Messung adaptierte Vorsteuerung	0.599 (1.227)	0.411 (1.284)
Mit ext. Messung an dieser Bahn adapt. Vor.	0.514 (1.043)	0.336 (1.097)
Ausführung der Schmid'schen Bahn an einem anderen Roboter		
Ohne Vorsteuerung	0.768 (5.473)	0.584 (4.891)
Mit ext. Messung adaptierte Vorsteuerung	0.309 (1.709)	0.306 (1.665)

Tabelle 6.7: Bahnfehler bei unterschiedlicher Messung der Ist-Bahn des Kuka KR6/1 Roboters (Messung der Motorbewegungen durch die Encoder und Berechnung der kartesischen Positionen bzw. direkte Messung der tatsächlichen Armpositionen mit dem Dynalog System. Alle Bahnfehler sind als Mittelwerte in mm angegeben (Maximalwerte in Klammern). Die allgemeine Vorsteuerung wurde aufgrund von Encoder-Messungen an Bahn 3a trainiert.)

Bahnfehler bei Messung durch	Encoder	Dynalog
Ohne Vorsteuerung	0.520 (4.022)	0.362 (3.597)
Allgemeine Vorsteuerung	0.124 (0.995)	0.425 (2.978)
1. mit ext. Messung adaptierte Vorsteuerung	0.377 (3.019)	0.174 (1.164)
2. mit ext. Messung adaptierte Vorsteuerung	0.339 (2.451)	0.165 (0.760)
3. mit ext. Messung adaptierte Vorsteuerung	0.338 (2.349)	0.169 (0.691)

Tabelle 6.8: Bahnfehler bei unterschiedlicher Messung der Ist-Bahn des Kuka KR15/2 Roboters (Messung der Motorbewegungen durch die Encoder und Berechnung der kartesischen Positionen bzw. direkte Messung der tatsächlichen Armpositionen mit dem Dynalog System. Alle Bahnfehler sind als Mittelwerte in mm angegeben (Maximalwerte in Klammern). Die allgemeine Vorsteuerung wurde aufgrund von Encoder-Messungen an Bahn 3a trainiert. Das Geschwindigkeitsprofil ist aufgrund der anderen Roboterkomponenten gegenüber Tabelle 6.7 leicht modifiziert. Daher sind die Ergebnisse nur eingeschränkt vergleichbar.)

Nach Punkt 5, also nach Reduktion der armseitigen Bahnfehler der Schmid'schen Bahn, sind die armseitigen Kennwerte beider Bahnen verbessert, die motorseitigen Werte sind leicht verschlechtert. Das Verhalten bei der Schmid'schen Bahn selbst ist zufrieden stellend (siehe Abbildungen 6.3 und 6.4).

Der Test bei einem anderen Roboter (siehe auch Abb. 6.5) zeigt jedoch, dass

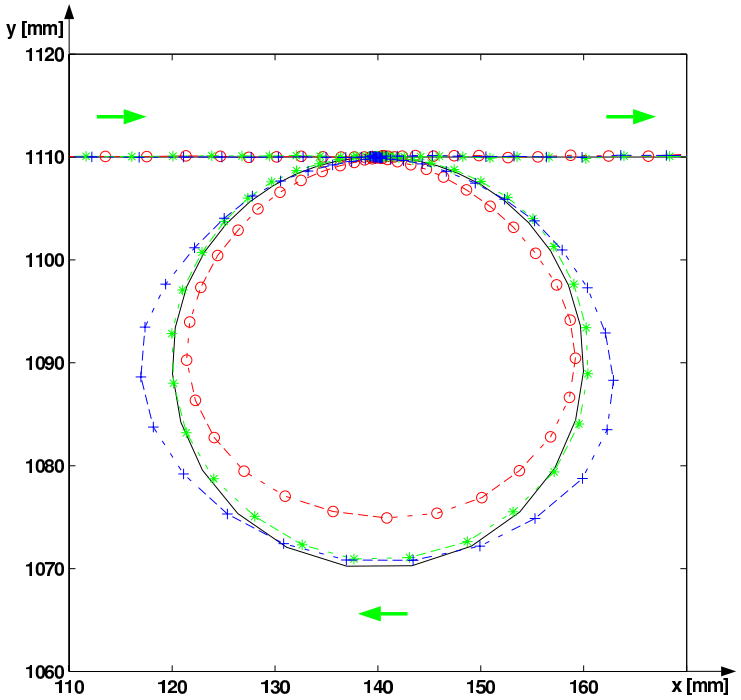


Abbildung 6.3: Ausführung des Kreises der Schmid'schen Bahn beim Kuka KR6/1 Roboter mit verschiedenen Reglern (o = ohne, + = allgemeine Vorsteuerung nach Punkt 1, * = mit externer Messung adaptierte Vorsteuerung nach Punkt 5, schwarz durchgezogen = Soll-Bahn)

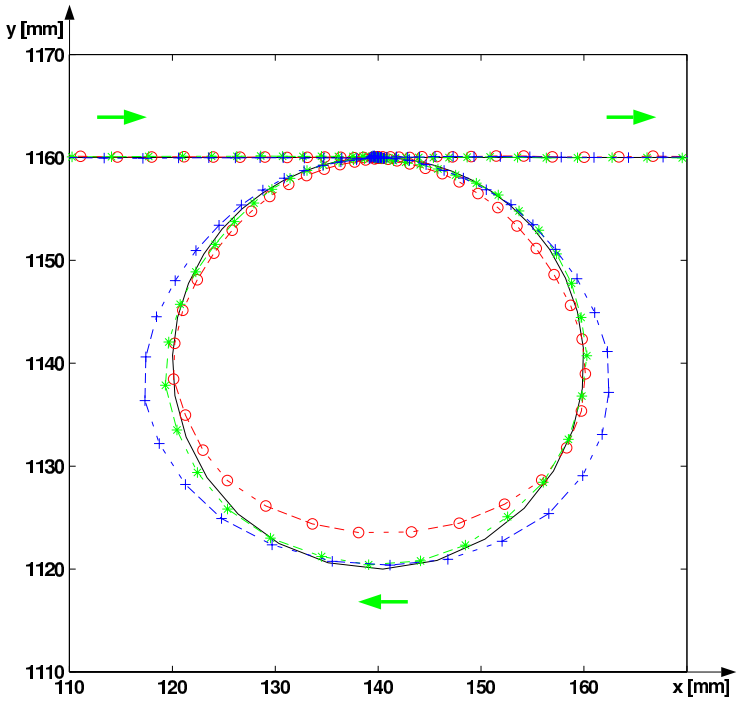


Abbildung 6.4: Ausführung des Kreises der Schmid'schen Bahn beim Kuka KR15/2 Roboter mit verschiedenen Reglern (○ = ohne, + = allgemeine Vorsteuerung nach Punkt 1, ∗ = mit externer Messung adaptierte Vorsteuerung nach Punkt 5, schwarz durchgezogen = Soll-Bahn)

das Trainingsergebnis weniger gut übertragbar ist als eine motorseitig trainierte Vorsteuerung. Das liegt an der Nachgiebigkeit, die nach Tabelle 6.7 zwischen Robotern gleichen Typs mehr schwankt als das durch die Kaskadenregelung beeinflusste motorseitige Verhalten. Dies ist durch Vergleich der letzten beiden Zeilen der Tabelle mit der ersten und dritten Zeile erkennbar. Die Unterschiede zwischen Soll-Bahn und antriebsseitig gemessener Ist-Bahn sind fast identisch, die Differenzen gegenüber den abtriebsseitig gemessenen Ist-Werten unterscheiden sich bei den beiden Robotern aber deutlich, bei adaptierter Vorsteuerung sogar um 50 % (Mittelwert).

Die Übertragung der an der Schmid'schen Bahn mit externer Messanordnung trainierten Vorsteuerung auf andere Bahnen ist möglich (siehe Zeile 6 von Tabelle 6.7). Ein noch besseres Verhalten ergibt sich natürlich, wenn Trainingsbahn und Testbahn identisch sind (siehe Zeile 7).

Zum Training einer Vorsteuerung für beliebige Bahnen muss man, wie schon in den vorangegangenen Abschnitten, Bahnen mit guter Anregung verwenden. Dies ist für Punkt 1 sicherlich durch Bahn 3a gegeben, während die Schmid'sche Bahn dazu weniger geeignet wäre. Zur Erfassung der Nachgiebigkeit in Punkt

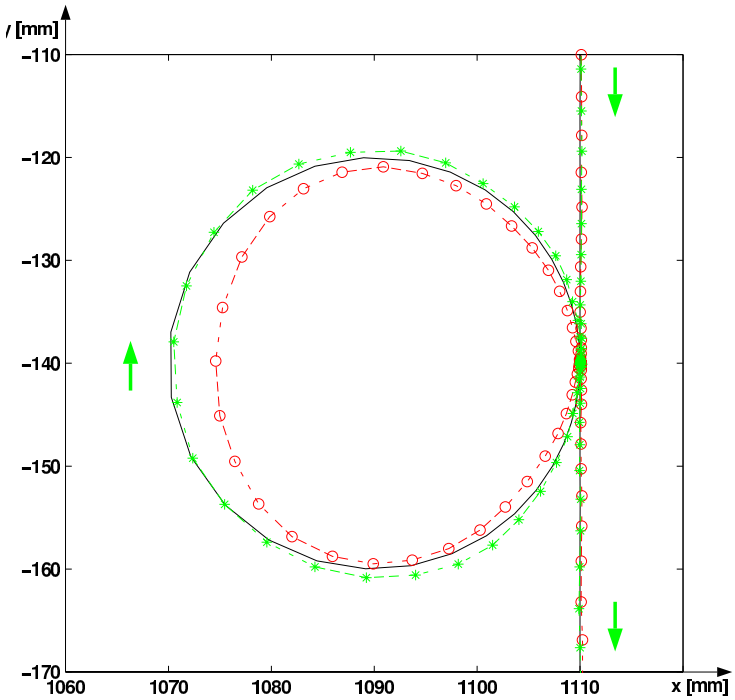


Abbildung 6.5: Ausführung des Kreises der Schmid'schen Bahn bei einem anderen Kuka KR6/1 Roboter (○ = ohne, ∗ = mit externer Messung adaptierte Vorsteuerung, schwarz durchgezogen = Soll-Bahn. Aus Platzgründen ist Achse 1 gegenüber Abb. 6.3 um 90° gedreht, was die Dynamik aber nicht beeinflusst.)

3 ist dagegen die Schmid'sche Bahn besser als Bahn 3a.

Die Verallgemeinerung der Vorsteuerung ist möglich, weil auch die Nachgiebigkeit einigermaßen linear, also auch ortsunabhängig, wirkt. Diese Linearität ist aber sicherlich schlechter als bei rein motorseitiger Betrachtung. In Abschnitt 3.2.1 wurde festgestellt, dass die wirksamen Trägheitsmomente nahezu ortsunabhängig sind, da die rotierenden Teile der Motoren wegen der Gewichtung mit dem Quadrat des Übersetzungsverhältnisses einen großen Anteil haben. Für die die Nachgiebigkeit beeinflussende Armträgheit gibt es keinen solchen konstanten Anteil. Daher sind mit externen Messwerten adaptierte Vorsteuerungen in größerem Abstand von der Trainingstrajektorie ggf. unbrauchbar.

6.1.4 Bewertung

Bei allen getesteten Robotertypen kann sowohl die mittlere als auch die maximale Bahngenauigkeit bei motorseitiger Messung stark erhöht werden. Auch schnelle Bahnen, die ohne Eingriff Bahnfehler im Millimeterbereich haben, wei-

sen mit Vorsteuerung nur noch mittlere Bahnfehler von etwa 0.2 mm und maximale Bahnfehler von etwa 1 mm auf. Dies gilt nach Training an einer oder mehreren geeigneten Bahnen nahezu bahnunabhängig im Arbeitsraum.

Die tatsächliche Genauigkeit, die durch externe Messung verifiziert wird, ist zunächst nicht so hoch. Durch Adaption aufgrund der externen Messung lassen sich die motorseitig erreichten Güten aber auch real erreichen. Allerdings schränkt die tatsächliche Verallgemeinerungsfähigkeit auf andere Bahnen oder Roboter die Anwendung einmal trainierter Vorsteuerungen auf andere Aufgaben ein.

Die in der Praxis erreichbare Genauigkeit kann am Vergleich von gleichartigen Robotern abgeschätzt werden. In der Praxis wird es nicht möglich sein, jeden Roboter bei jeder neuen Bahn mit einem externen Messsystem zu versehen und die Vorsteuerung entsprechend abzuändern. Stattdessen wird die Vorsteuerung entweder bei der Inbetriebnahme eines Roboters bestimmt und dann für alle vorkommenden Bahnen verwendet, oder für jede neue Bahn wird im Labor die geeignete Vorsteuerung bestimmt, die dann zusammen mit den Bahndaten an die Produktion übertragen wird. In beiden Fällen ist, wie Tabelle 6.7 zeigt, mit einer Verschlechterung gegenüber dem optimalen Verhalten zu rechnen. Gegenüber dem Fall ohne Vorsteuerung bleibt aber noch eine Halbierung der Bahnfehler.

Eine bessere Übertragbarkeit ist ggf. möglich, wenn vom Zielsystem, also von dem System, an dem die Vorsteuerung eingesetzt werden soll, Messwerte vorliegen. So können die Rückwirkungen einer z. B. wegen unterschiedlicher Reibung anderen Nachgiebigkeit unter Umständen durch die motorseitigen Achswerte beobachtbar sein. Diese Messwerte werden wahlweise aufgezeichnet und beim Entwurf der Vorsteuerung berücksichtigt, oder die Struktur der Vorsteuerung (Regelung) wird derart erweitert, dass auch Messwerte online verarbeitet werden.

Aus der Übertragbarkeit der Vorsteuerungen von einem Roboter zum anderen kann man auch das bestmögliche Verhalten bei modellbasierter Regelung abschätzen. Dabei wird die Regelung bei einem Roboter mit nominalen Parametern zufrieden stellend arbeiten, ohne Verwendung von Messwerten des Zielsystems werden die Bahnfehler an anderen Robotern aber um die gleiche Größenordnung höher liegen, wie bei den vorliegenden Experimenten. Dabei ist man nach Untersuchung von zwei ungefähr gleich alten Robotern sicherlich

nicht in der Lage, die Bandbreite der Unterschiede seriös abzuschätzen.

Schließlich muss noch festgestellt werden, dass die verwendeten Testbahnen keine Schwingung des Roboterarmes angeregt haben. Solche Schwingungen lassen sich mit dem vorgeschlagenen Ansatz aufgrund der verwendeten Schnittstelle geringer Bandbreite nicht kompensieren.

6.2 Verbesserungspotential durch Neuronale Netze

In Abschnitt 6.1.1 wurden Kopplungen bei einem Manutec r2 Roboter festgestellt. Achse 4 wird durch die Beschleunigungen von Achse 1 beeinflusst. Dies wird in diesem Abschnitt durch ein Neuronales Netz kompensiert.

Tabelle 6.2 lässt vermuten, dass außer bei Achse 4 auch bei Achse 5 unberücksichtigte Einflüsse wie Kopplungen existieren, da dort höhere Fehler als bei den Grundachsen auftreten. Dagegen kann der hohe Fehler von Achse 6 aufgrund der symmetrischen Anordnungen nicht von Kopplungen stammen. Daher werden in diesem Abschnitt Neuronale Netze nach Abschnitt 5.5.3 für Achse 4 und Achse 5 implementiert (Gleichungen (5.60) bzw. (5.61)).

Dabei werden insgesamt drei Versuche unternommen. Zuerst wird an Bahn 3 trainiert und getestet, um die Repräsentationsfähigkeit des Netzes zu überprüfen. Dann wird die Verallgemeinerung nach Training an Bahn 1 und Bahn 3 getestet. Schließlich wird die Verallgemeinerung in einem engeren Bereich um die Trainingsbahn untersucht, da dies für die typische Anwendung am wichtigsten ist, nämlich für die sensorische Feinkorrektur von offline nur grob geplanten Bahnen.

Das Netztraining erfolgt durch EKFNet (Anhang B). Die Rechenzeiten werden im folgenden maschinenunabhängig in Epochen angegeben, wobei bei der Trainingsbahn 3 das Training einer Epoche (196 Schritte) auf einem mit 50 MHz getakteten R4000 Prozessor beim Netz für Achse 4 etwa 0.7 s und beim Netz für Achse 5 etwa 2.2 s dauert. Bei der kürzeren Bahn 1 (78 Schritte) betragen die Rechenzeiten etwa 0.3 s und 0.9 s. Die Zeiten werden bei gemeinsamem Training mehrerer Trainingsbahnen einfach addiert.

Als erstes Experiment wird nun das Netz an einer Bahn trainiert, die gemäß Tabelle 6.1 vortrainiert wurde. Dabei wird zunächst Bahn 3 einmal mit linearer

Iteration	Positionsfehler	Netzfehler		Rechenzeit in
		vor Training	nach Training	Epochen
0	0.507 mrad	0.461 mrad	0.212 mrad	9
1	0.241 mrad	0.181 mrad	0.090 mrad	7
2	0.149 mrad	0.084 mrad	0.046 mrad	10
3	0.138 mrad	0.071 mrad	0.035 mrad	7
4	0.127 mrad	0.062 mrad	0.032 mrad	10
5	0.106 mrad	0.042 mrad	0.022 mrad	1
6	0.101 mrad	0.035 mrad	0.022 mrad	1
7	0.095 mrad	0.029 mrad	-	-

Tabelle 6.9: Verringerung des mittleren Positionsfehlers von Achse 4 durch Training eines Neuronalen Netzes

Vorsteuerung abgefahren. Die dabei entsprechend Abschnitt 5.2.1 bestimmten Stellgrößenänderungen werden nun in den Netzen für die beiden Achsen trainiert, bis eines der Abbruchkriterien greift. Dann wird mit der so entstandenen neuen nichtlinearen Vorsteuerung die Bahn wieder abgefahren. Die daraus bestimmten Stellgrößenänderungen werden wieder trainiert. In gleicher Weise wird so lange fortgefahren, bis sich keine weitere Verbesserung mehr erreichen lässt.

Tabelle 6.9 und Tabelle 6.10 zeigen dabei neben dem mittleren Positionsfehler auch noch die mittleren Netzfehler nach Gleichung (5.62 vor und nach dem Training. Der Netzfehler vor dem Training ist etwas kleiner als der Positionsfehler, da bei der Bestimmung der optimalen Stellgrößen u_{neu} gefiltert wird. Die Dif-

Iteration	Positionsfehler	Netzfehler		Rechenzeit in
		vor Training	nach Training	Epochen
0	0.427 mrad	0.249 mrad	0.245 mrad	10
1	0.417 mrad	0.240 mrad	0.176 mrad	10
2	0.371 mrad	0.185 mrad	0.127 mrad	10
3	0.334 mrad	0.138 mrad	0.107 mrad	10
...	...	...	...	...
12	0.242 mrad	0.046 mrad	0.032 mrad	10
13	0.234 mrad	0.043 mrad	0.030 mrad	8
14	0.229 mrad	0.039 mrad	0.030 mrad	4
15	0.227 mrad	0.035 mrad	0.030 mrad	2
16	0.224 mrad	0.035 mrad	0.029 mrad	2
17	0.223 mrad	0.038 mrad	0.029 mrad	1
18	0.222 mrad	0.033 mrad	0.028 mrad	1
19	0.227 mrad	0.034 mrad	0.028 mrad	1
20	0.223 mrad	0.030 mrad	-	-

Tabelle 6.10: Verringerung des mittleren Positionsfehlers von Achse 5 durch Training eines Neuronalen Netzes

ferenz entspricht ungefähr den überhaupt erreichbaren Positionsfehlern. Diese lassen sich durch Training der Bahnsteuerung (Abschnitt 6.4) abschätzen. Dabei wird die maximale Genauigkeit von 0.063 mrad bei Gelenk 4 bzw. 0.194 mrad bei Gelenk 5 erreicht.

Zu Beginn des Trainings spiegelt sich die Abnahme des Netzfehlers durch das Training einer Trajektorie in einer etwa gleichgroßen Reduktion des Positionsfehlers wieder, der beim Abfahren mit dem so trainierten Netz gemessen wird (Beispiel: 0.461 - 0.212 $\approx$ 0.507 - 0.241). Der Netzfehler vor dem Training der nächsten Trajektorie ist dem Fehler nach dem letzten Training ähnlich. (Beispiel: 0.212 $\approx$ 0.181).

Das lässt vermuten, dass die Eingänge des Neuronalen Netzes das Problem vollständig beschreiben und dass die Verallgemeinerung für die stochastischen Unterschiede zwischen den einzelnen Trajektorien ausreicht. Andernfalls, in dem Extremfall, dass die Eingänge nichts mit den Positionsfehlern zu tun haben, dürfte die Reduktion des Netzfehlers einer Trajektorie keine Auswirkung auf den Positionsfehler oder den daraus resultierenden Netzfehler der folgenden Trajektorie haben.

Gegen Ende des Trainings werden die stochastischen Unterschiede zwischen den Trajektorien dominant. Das Training einer Trajektorie verschlechtert also das vorherige Lernen ähnlicher Datensätze. Die Reduktion des Positionsfehlers einer neuen Trajektorie, die ähnlich zu allen vorherigen ist, fällt dadurch nicht so hoch wie erwartet aus. Tabelle 6.9 und Tabelle 6.10 zeigen aber, dass der Netzfehler bis auf die Abbruchschranke von 0.030 mrad reduziert wird. Die Daten lassen sich also mit den vorgegebenen Netzstrukturen trainieren.

Die Abnahme des mittleren Positionsfehlers ist in Abb. 6.6 auch grafisch dargestellt. Zur Konvergenzgeschwindigkeit lässt sich bemerken, dass das Netz für Achse 4 innerhalb von 32 Sekunden einschwingt. Bei Achse 5 werden wegen der höheren Zahl an Eingängen und der komplizierteren Bewegung (Ohne Vorsteuerung beträgt der Positionsfehler dort 71 mrad !) etwa 5 Minuten Rechenzeit des R 4000 benötigt.

Bei beiden Achsen ist also eine deutliche Verbesserung des Positionsfehlers festzustellen. Die Werte der Steuerung von 0.063 mrad bzw. 0.194 mrad werden bei den programmierten Abbruchkriterien nicht ganz erreicht. Abb. 6.7 und Abb. 6.8 zeigen aber, dass die Rampen bei Achse 4 und die deterministischen Anteile des Positionsfehlers bei Achse 5 stark gedämpft sind.

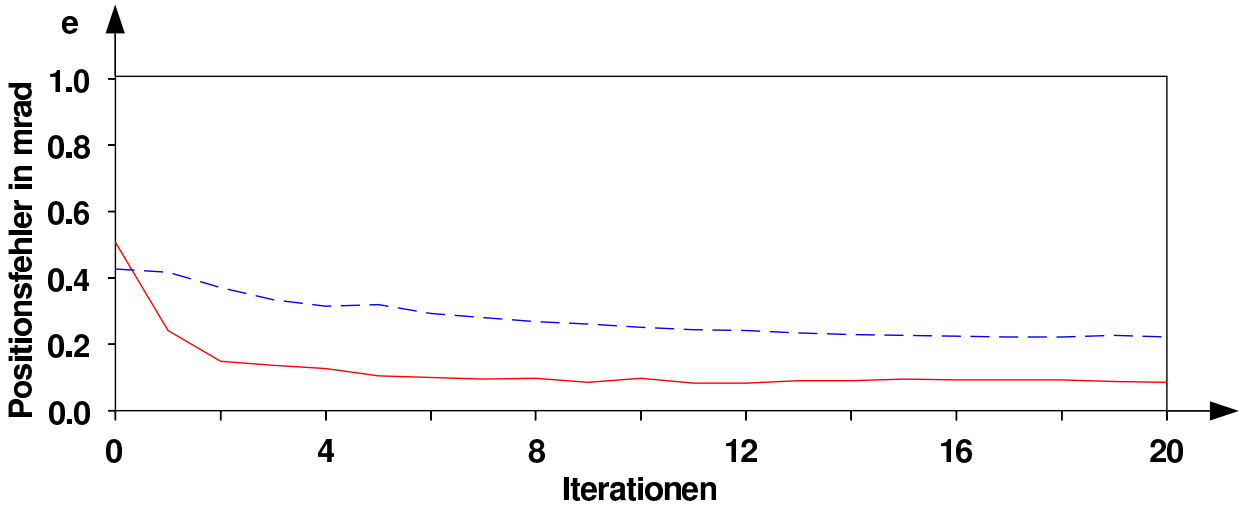


Abbildung 6.6: Abnahme der mittleren Positionsfehler von Achse 4 (rot durchgezogen) und Achse 5 (blau gestrichelt) beim Training des Neuronalen Netzes an Bahn 3

Diese Verbesserung gilt aber nur an der Trainingsbahn selbst. Bei anderen Bahnen ergibt sich sogar eine Verschlechterung. Aus diesem Grund wird versucht, durch Training von zwei grundverschiedenen Bahnen den Arbeitsraum zumindest einigermaßen zu erfassen. Das ist vergleichbar mit Abschnitt 6.1.2, in dem als Alternative die Vorsteuerung durch zwei unterschiedliche Trainingsbahnen bestimmt wurde.

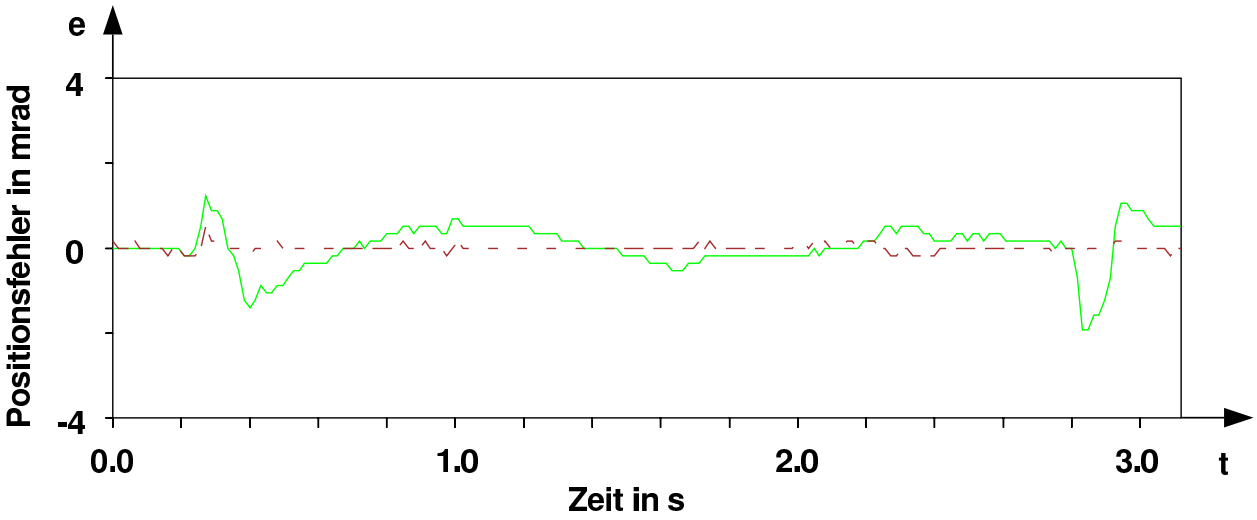


Abbildung 6.7: Positionsfehler von Achse 4 mit (braun gestrichelt) und ohne (grün durchgezogen) Neuronalem Netz

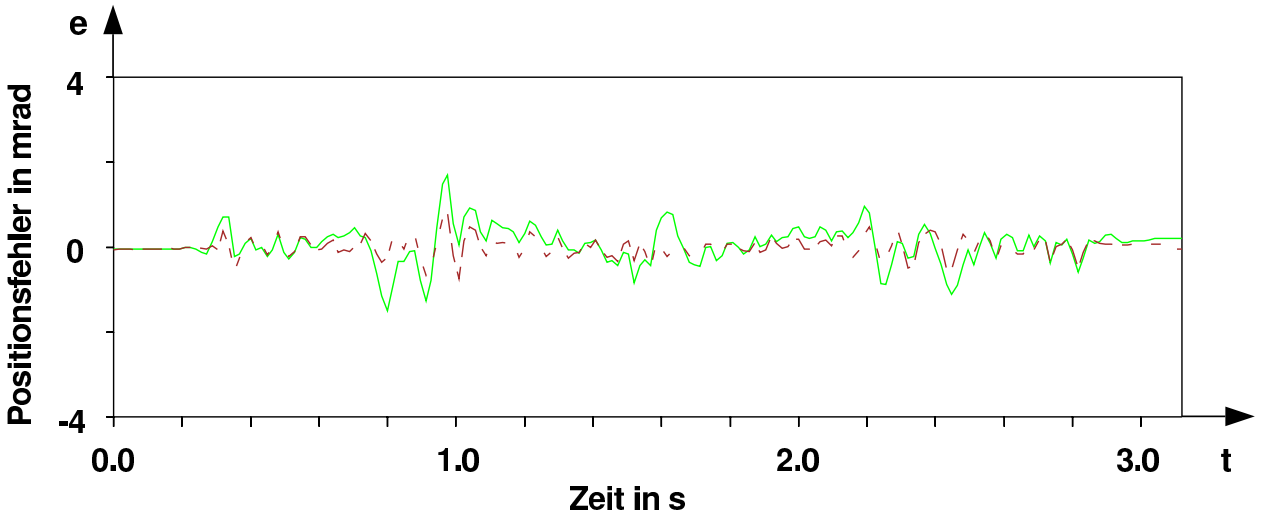


Abbildung 6.8: Positionsfehler von Achse 5 mit (braun gestrichelt) und ohne (grün durchgezogen) Neuronalem Netz

Das Training an den beiden Bahnen, die auch für Tabelle 6.5 abwechselnd abgefahren wurden, ist in Abb. 6.9 dokumentiert. Dabei wird das Netz jeweils trainiert, nachdem beide Bahnen abgefahren wurden. Die Bahnen müssen jeweils zweimal abgefahren werden, um reproduzierbare Ergebnisse zu erhalten. Der jeweils erste Lauf einer neuen Bahn führt nämlich zu einem etwas anderen Ergebnis, da die Anfangsposition offensichtlich nur im Rahmen der Auflösung

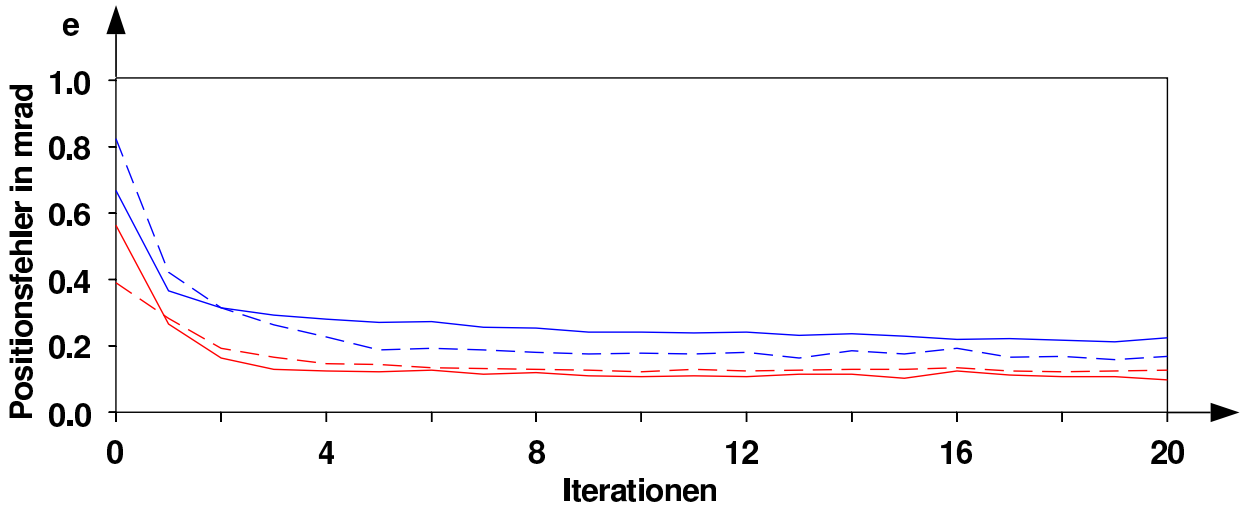


Abbildung 6.9: Abnahme der mittleren Positionsfehlers der beiden Trainingsbahnen beim Training des Neuronalen Netzes (Bahn 1 gestrichelt, Bahn 3 durchgezogen, Gelenk 4 rot (unten), Gelenk 5 blau (oben))

Bahn	Achse 4		Achse 5	
Bahn	ohne Netz	mit Netz	ohne Netz	mit Netz
1	0.796	0.168	0.349	0.128
2	0.710	1.472	0.411	1.104
3	0.528	0.097	0.483	0.224
4	0.246	1.427	0.353	1.397
5	0.138	1.529	0.455	1.880

Tabelle 6.11: Test des Positionsfehlers nach Training an Bahn 1 und Bahn 3

der Encoderwerte angefahren wird, die genaue Position also auch von der Geschwindigkeit abhängt.

Es zeigt sich, dass das Netz auch in dem Fall die Trainingsdaten mit geringem Fehler speichern kann. So dauert das Lernen zwar etwas länger (Rechenzeit für Achse 5 etwa 10 Minuten), die Positionsfehler erreichen aber für Bahn 3 die gleichen Werte wie in Abb. 6.6.

Die Trainingsdaten der beiden Bahnen widersprechen sich also nicht. Denkbar wäre auch, dass bei vorgegebener Netzstruktur zwar beide Bahnen einzeln lernbar sind, dass das Netz bei gemeinsamem Training von verschiedenen Bahnen aber keine Darstellung mit geringem Fehler mehr erzeugen kann. In dem Fall müsste die Netzstruktur erweitert werden.

Beim Test an untrainierten Bahnen ist aber auch nach diesem Training an zwei unterschiedlichen Bahnen eine starke Verschlechterung bei beiden Achsen messbar (siehe Tabelle 6.11), selbst wenn die Beschleunigung am Start und das Abbremsen am Ende der Bahn, also die Problemstellen für Achse 4, mit den Trainingsbahnen übereinstimmen.

Abb. 6.10 zeigt, dass die erhöhten Positionsfehler bei Bahn 4 über die ganze Bahn verteilt sind, also auch über den mittleren Kreis in Abb. C.4, obwohl dieser, abgesehen von der Geschwindigkeit, identisch mit Bahn 3 ist.

Daran sieht man, dass der Netzeingang auch bei ähnlichen Bahnen stark unterschiedlich sein kann. Dies gilt sowohl bei veränderter Geschwindigkeit als auch, wegen der absoluten Achswerte im Netzeingang, bei translatorischer Verschiebung der Bahn. Es ist daher zu vermuten, dass es keine Kombination von verschiedenen Trainingsbahnen gibt, die ein globales Training für den ganzen Arbeitsraum des Roboters ergeben.

Auf der anderen Seite macht es wenig Sinn, ein Netz nur für eine Bahn zu

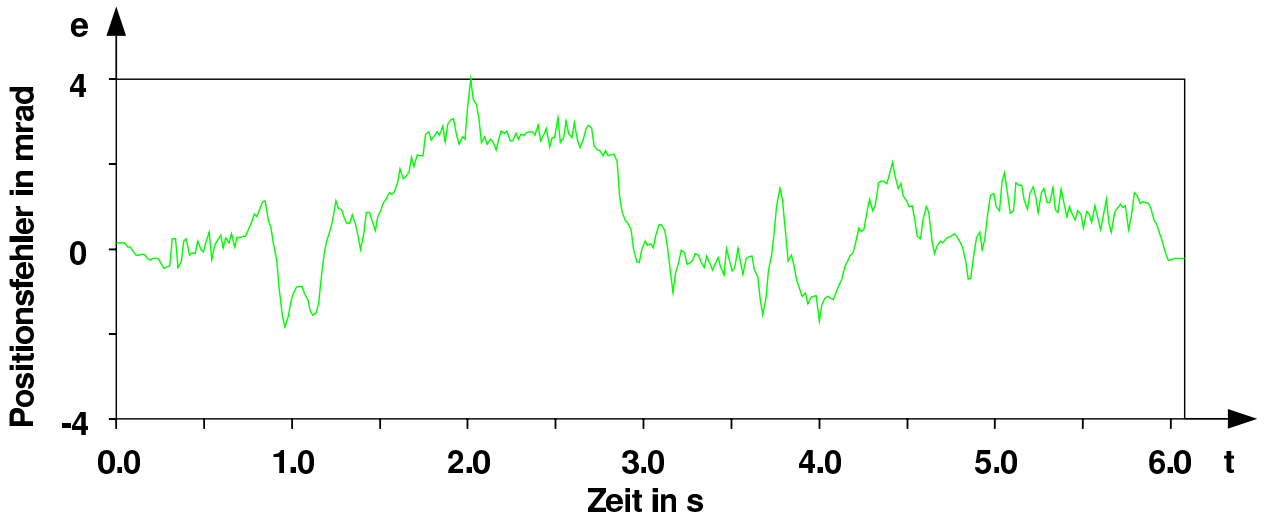


Abbildung 6.10: Positionsfehler von Achse 5 beim Abfahren von Bahn 4 nach Training des Neuronalen Netzes an Bahn 1 und Bahn 3

bestimmen, da dafür das Trajektorienlernen einer reinen Steuerung einfacher und genauer ist. Wegen der mangelnden Verallgemeinerung bietet sich nur eine Anwendungsmöglichkeit. Und zwar sind dies grob definierte Bahnen, die im Detail erst online, z. B. sensorabhängig festgelegt werden.

Zum Test wird nun Bahn 3 trainiert, wobei zum Training des Netzes nicht die Nominalbahn selbst abgefahren wird, sondern 6 Bahnen, die um den halben Radius von 100 mm in den kartesischen Achsenrichtungen verschoben sind. Danach werden die entsprechend Abschnitt 5.2.1 bestimmten Stellgrößenänderungen aller 6 Bahnen mit den zugehörigen Netzeingangsdaten zu einer gemeinsamen Trainingsdatei zusammengefasst und trainiert und das resultierende Netz wird an der Nominalbahn getestet. Dies wird ohne explizites Training der Nominalbahn einige Male wiederholt.

Tabelle 6.12 und Tabelle 6.13 zeigen das Resultat. Das Training verallgemeinert so gut, dass auch die Nominaltrajektorie wesentlich verbessert wird. Die Positionsfehler des direkten Trainings aus Tabelle 6.9 und Tabelle 6.10 werden allerdings nicht ganz erreicht. Aber die Ergebnisse liegen in der Größenordnung der Werte der linearen Vorsteuerung der übrigen Gelenke gemäß Tabelle 6.2. Die Aufgabe, die beiden bei linearer Vorsteuerung nicht zufrieden stellenden Achsen weiter zu verbessern, ist also erfüllt.

Das Training zeigt aber auch, insbesondere bei Achse 5, dass mit der vorge-

Iteration	Positionsfehler	Netzfehler		Rechenzeit in
		vor Training	nach Training	Epochen
0	0.546 mrad	0.489 mrad	0.231 mrad	3
1	0.262 mrad	0.199 mrad	0.090 mrad	2
2	0.148 mrad	0.079 mrad	0.041 mrad	10
3	0.127 mrad	0.055 mrad	0.030 mrad	6
4	0.122 mrad	0.041 mrad	0.028 mrad	1
5	0.115 mrad	0.040 mrad	0.027 mrad	1
6	0.115 mrad	0.037 mrad	0.027 mrad	1
7	0.117 mrad	0.034 mrad	0.025 mrad	1
8	0.120 mrad	0.032 mrad	0.025 mrad	1
9	0.108 mrad	0.029 mrad	-	-

Tabelle 6.12: Verringerung des Positionsfehlers von Achse 4 beim Abfahren der Nominalbahn nach Training des Neuronalen Netzes in der Umgebung

gebenen Netzkonfiguration die Trainingsdaten nicht fehlerfrei zu lernen sind. So bringt die Vorsteuerung auch bei den trainierten Bahnen im Mittel keine geringeren Positionsfehler als bei der Nominaltrajektorie. Die Alternative eines größeren Netzes mit ggf. zusätzlichen Eingängen erscheint aber nicht nötig, zumal dies die Rechenzeit von insgesamt etwa 24 Minuten noch erhöhen würde.

Festzuhalten bleibt, dass ein Neuronales Netz bei Training einer Bahn mit Umgebung so gut verallgemeinert, dass der Positionsfehler gegenüber einer linearen Vorsteuerung auch dann geringer ist, wenn beim Test die Bahn leicht verändert wird. Dabei sind, sofern trainiert, Abweichungen von 50 mm problemlos möglich. Dadurch ist gezeigt, dass es sinnvoll ist, Bahnen, die online

Iteration	Positionsfehler	Netzfehler		Rechenzeit in
		vor Training	nach Training	Epochen
0	0.453 mrad	0.275 mrad	0.202 mrad	10
1	0.378 mrad	0.198 mrad	0.154 mrad	10
2	0.354 mrad	0.150 mrad	0.124 mrad	10
3	0.326 mrad	0.126 mrad	0.108 mrad	10
4	0.309 mrad	0.112 mrad	0.099 mrad	10
5	0.299 mrad	0.126 mrad	0.107 mrad	10
6	0.298 mrad	0.115 mrad	0.097 mrad	10
7	0.290 mrad	0.103 mrad	0.093 mrad	10
8	0.289 mrad	0.092 mrad	0.084 mrad	10
9	0.278 mrad	0.083 mrad	0.076 mrad	10
10	0.272 mrad	0.077 mrad	-	-

Tabelle 6.13: Verringerung des Positionsfehlers von Achse 5 beim Abfahren der Nominalbahn nach Training des Neuronalen Netzes in der Umgebung

sensorabhängig modifiziert werden, nicht gesteuert abzufahren, sondern vorgesteuert mit einem linearen gelenkinternen Vorfilter und einem Neuronalen Netz zur Kompensation der gelenkübergreifenden Kopplungen.

Zum Ende dieses Abschnittes kann noch bemerkt werden, dass die hier durchgeführten Trainingsläufe in [151] auch mit anderen Trainingsverfahren ausgeführt wurden. Dabei zeigte sich, dass das verwendete erweiterte Kalman Filter geeignet ist.

6.3 Einfluss von Kontaktkräften

Die neben dem Ansatz Neuronaler Netze andere Erweiterung der normalen Positionsvorsteuerung nach Gleichung (5.44) betrifft die Kraftvorsteuerung, also den Regleransatz nach Gleichung (5.53).

Zur Demonstration dieses Einflusses wird an dem in Abb. 3.17 gezeigten Teil der Kontur eine Kraftvorsteuerung für den Manutec r2 Roboter trainiert. Das komplette Training ist in Tabelle 6.14 dokumentiert. Es beginnt mit der Adaption der Positionsvorsteuerung, die ohne Kontakt an Bahn 1 trainiert wird. Dann wird nach Abschnitt 6.6 die Bahn ertastet, für die minimale Abweichungen vom Kraftsollwert von 10 N auftreten. Dies wird später wiederholt, da aufgrund von Verbiegungen des Stabes die mittlere Kraft leicht driftet. An dieser Bahn wird die Kraftvorsteuerung adaptiert. Dadurch wird auch die Positionsvorsteuerung leicht verändert.

Danach wird die gesamte Kontur aus Abb. 3.1 nach Abschnitt 6.6 ertastet und abgespeichert. Dabei werden neben den Gelenkpositionen auch die aufgrund des Kontakts zusätzlich auftretenden Gelenkmomente nach den Gleichungen (5.50) bis (5.52) berechnet und abgelegt.

Tabelle 6.15 vergleicht die Positionsfehler dieser Bahn mit dem zum Training benutzten Ausschnitt. Dabei wird jeweils zwischen dem Positionsfehler beim Ertasten der Kontur und bei der späteren Wiederholung unterschieden, da beim Ertasten keine Vorsteuerung möglich ist (vergl. Abschnitt 6.6).

Zusätzlich zum Abfahren der Bahn mit konstanter Geschwindigkeit wird für die ertastete Bahn offline ein Geschwindigkeitsprofil berechnet, durch das die schnellstmögliche Bewegung ausgeführt wird, also die Bewegung, bei der die

Iteration	Bewegung	Positionsfehler	Kraftfehler	Adaption von
0	Bahn 1 mit 3 mm/Schritt	13.612 mm	-	Modell
1	Bahn 1 mit 6 mm/Schritt	23.127 mm	-	Vorsteuerung
2	Bahn 1 mit 6 mm/Schritt	2.886 mm	-	Vorsteuerung
3	Bahn 1 mit 6 mm/Schritt	0.586 mm	-	Vorsteuerung
4	Bahn 1 mit 6 mm/Schritt	0.264 mm	-	Vorsteuerung
5	Bahn 1 mit 6 mm/Schritt	0.223 mm	-	Vorsteuerung
6	Bahn 1 mit 6 mm/Schritt	0.213 mm	-	Vorsteuerung
7	Bahn 1 mit 6 mm/Schritt	0.204 mm	-	Vorsteuerung
8	Bahn 1 mit 6 mm/Schritt	0.204 mm	-	Vorsteuerung
9	Bahn 1 mit 6 mm/Schritt	0.205 mm	-	Vorsteuerung
10	Bahn 1 mit 6 mm/Schritt	0.209 mm	-	Vorsteuerung
-	Bahn 1 mit 6 mm/Schritt	0.205 mm	-	keine Adaption
11	Bahn ertasten	0.443 mm	0.816 N	Vorsteuerung
12	ertaste Bahn	0.117 mm	0.409 N	Vorsteuerung
13	ertaste Bahn	0.102 mm	0.661 N	Vorsteuerung
14	ertaste Bahn	0.102 mm	0.891 N	Vorsteuerung
15	ertaste Bahn	0.111 mm	1.124 N	Vorsteuerung
16	Bahn ertasten	0.422 mm	0.763 N	Vorsteuerung
17	ertaste Bahn	0.101 mm	0.278 N	Vorsteuerung
18	ertaste Bahn	0.095 mm	0.407 N	Vorsteuerung
19	ertaste Bahn	0.110 mm	0.638 N	Vorsteuerung
20	Bahn ertasten	0.441 mm	0.802 N	Vorsteuerung
-	ertaste Bahn	0.098 mm	0.268 N	keine Adaption

Tabelle 6.14: Ablauf des Trainings der Kraftvorsteuerung (Das Ertasten und Wiederholen der Bahn findet bei 0.6 mm/Schritt über 5 s statt)

zulässigen Beschleunigungen während der ganzen Trajektorie erreicht werden. Dabei wird eine mittlere Geschwindigkeit von 4.2 mm/Schritt bei einem Spitzenwert von 11.5 mm/Schritt erreicht. Diese Bahn würde ohne Vorsteuerung zur Überschreitung der zulässigen Kräfte führen.

Der Vergleich von Spalte 4 und Spalte 7 zeigt, dass die Verallgemeinerung der Kraftvorsteuerung ausreichend ist. Im trainierten Bereich ist aufgrund der stärkeren Krümmungen sogar ein schlechterer Positionsfehler aufgetreten. Man kann also davon ausgehen, dass auch die Kraftvorsteuerung einigermaßen orts- und richtungsunabhängig trainiert wurde.

Die beiden letzten Spalten zeigen, dass die Kraftvorsteuerung auch bei wesentlich höheren Geschwindigkeiten als während des Trainings den Positionsfehler deutlich verkleinern kann.

Die Verbesserung durch die Kraftvorsteuerung ergibt sich insbesondere in den

Achse	vorderer Teil 0.6 mm/Schritt			ganze Kontur 0.6 mm/Schritt			ganze Kontur max. Geschw.	
	T	P	P & K	T	P	P & K	P	P & K
1	0.664	0.051	0 051	0.505	0.052	0.052	0.319	0.323
2	0.953	0.066	0.067	0.668	0.062	0.062	0.526	0.513
3	1.456	0.128	0.133	0.889	0.114	0.115	0.534	0.541
4	0.427	0.391	0.109	0.275	0.267	0.113	0.984	0.468
5	0.751	0.274	0.199	0.397	0.189	0.154	1.054	0.864
6	0.770	0.469	0.500	0.666	0.478	0.477	1.060	1.092
kart.	0.445	0.187	0.097	0.296	0.122	0.080	0.522	0.377

Tabelle 6.15: Positionsfehler beim Abfahren der Kontur: T = Tasten der Bahn ohne Vorsteuerung, P = Abfahren mit Positionsvorsteuerung, P & K Abfahren mit Positions- und Kraftvorsteuerung (Achsfehler in mrad, kart. Fehler in mm)

„weichen“ Handgelenken, den Achsen 4 und 5. Auf Achse 6 wirken aufgrund des abrollenden Stiftes (vergl. Abb. 3.1) keine Kräfte.

Tabelle 6.15 zeigt, dass sich eine Kraftvorsteuerung auch dann lohnen kann, wenn der Krafteinfluss auf eine nicht vorgesteuerte Bahn relativ gering ist. Durch die starke Reduktion der Bahnfehler aufgrund der Positionsvorsteuerung werden die angreifenden Kontaktkräfte leicht zur dominanten Störgröße.

6.4 Erreichbare Genauigkeit bei wiederholten Bewegungen

In diesem Abschnitt wird die Bestimmung einer Bahnsteuerung nach Abschnitt 5.2.1 betrachtet. Die Adaption ist einerseits direkt möglich, also ausgehend von den der Soll-Trajektorie entsprechenden Bewegungsanweisungen, oder als Zusatz nach dem Abfahren mit Vorsteuerung, ausgehend von den vorgesteuerten Bewegungsanweisungen. Während in der Praxis meist die zweite Form gewählt wird, erfolgt die Erläuterung hier ohne vorherige Vorsteuerung, da die Vorsteuerung so direkt mit der Bahnsteuerung verglichen werden kann.

Die Experimente für diesen Abschnitt werden nur am Manutec r2 Roboter ausgeführt, da bei den Robotern der Firma Kuka schon bei Vorsteuerung eine Genauigkeit erreicht wird, die über der Messgenauigkeit der Verifikation liegt.

Tabelle 6.16 vergleicht die Konvergenzgeschwindigkeit beim Training der Bahnsteuerung von Bahn 3 mit 500 mm/s mit der Adaption der Vorsteuerung.

	Vorsteuerung	Bahnsteuerung
0	39.893 mm	39.908 mm
1	6.024 mm	6.249 mm
2	1.205 mm	1.266 mm
3	0.441 mm	0.380 mm
4	0.310 mm	0.215 mm
5	0.253 mm	0.175 mm
6	0.232 mm	0.156 mm
7	0.225 mm	0.152 mm
8	0.219 mm	0.143 mm
9	0.214 mm	0.144 mm
10	0.211 mm	0.137 mm

Tabelle 6.16: Abnahme der mittleren kartesischen Positionsfehler während des Trainings von Bahn 3 mit 500 mm/s

Die starke Verbesserung am Anfang der Adaption ist in beiden Fällen gleich. Das Endergebnis ist jedoch deutlich besser (siehe auch Abb. 6.11). Das kommt daher, dass bei der Vorsteuerung nur lineare Einflüsse und nur Wirkungen auf dieselbe Achse berücksichtigt werden. Dagegen ist bei der Bahnsteuerung auch eine an gleichen Bahnpunkten immer gleich auftretende Kopplung oder die ortsabhängige Trägheit iterativ ausgeglichen.

Tabelle 6.17 zeigt die Positionsfehler der einzelnen Achsen. Auffällig ist der Unterschied in Achse 4. Es zeigt sich deutlich, dass die Bahnsteuerung trotz des achsweisen Trainings auch systematische achsübergreifende Einflüsse wie

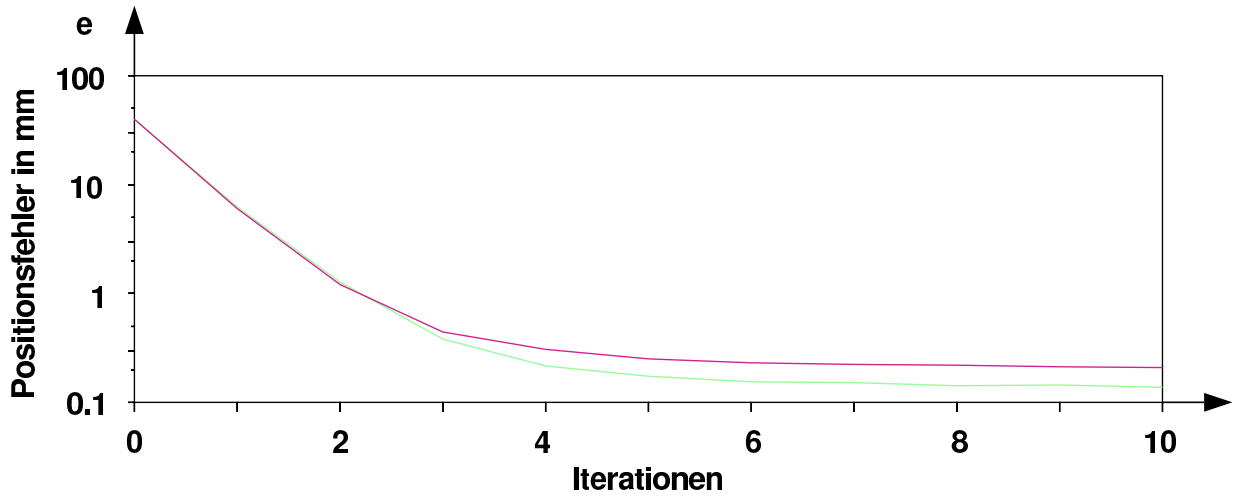


Abbildung 6.11: Abnahme der mittleren Positionsfehler beim Training von Vorsteuerung (violett (oben)) oder Bahnsteuerung (grün (unten))

	mit Vorsteuerung	mit Bahnsteuerung
Positionsfehler von Achse 1	0.313 mrad	0.280 mrad
Positionsfehler von Achse 2	0.185 mrad	0.109 mrad
Positionsfehler von Achse 3	0.311 mrad	0.123 mrad
Positionsfehler von Achse 4	0.499 mrad	0.068 mrad
Positionsfehler von Achse 5	0.457 mrad	0.192 mrad
Positionsfehler von Achse 6	1.040 mrad	0.388 mrad
kartesischer Positionsfehler	0.211 mm	0.137 mm

Tabelle 6.17: Vergleich der Genauigkeit des Manutec r2 Roboters nach Training einer Vorsteuerung bzw. Bahnsteuerung von Bahn 3 mit 500 mm/s

Kopplungen kompensieren kann. So werden die Werte von 0.097 mrad und 0.224 mrad, die nach Training an Bahn 1 und Bahn 3 mit Neuronalen Netzen für die Achsen 4 und 5 erreicht wurden, durch die Bahnsteuerung noch unterschritten.

Beim Training von langsamen Bahnen ist es sogar möglich, noch kleinere Positions- und Bahnfehler zu erreichen. Das wird in Tabelle 6.18 demonstriert. Hier liegen die Grenzen der Verbesserung nicht in den Möglichkeiten der Korrektur sondern in der Genauigkeit des Roboters. Diese wird begrenzt durch die Rundungsfehler bei der Umwandlung der am Host-Rechner kommandierten Gelenkwinkel in die roboterinterne Festkommadarstellung und durch die entsprechenden Quantisierungssprünge der Gelenkistwerte. Tabelle 6.18 zeigt, dass die erreichte Genauigkeit in der Größenordnung des Rundungsfehlers liegt. Letzterer wird dabei durch Vergleich der Kommandos mit den Rückmeldungen der RCM berechnet. Der Positionsfehler verschwindet also im Rahmen der Auflösung der Kommandos.

Die Tabellen 6.19 und 6.20 fassen die Bahnsteuerungen und Vorsteuerungen

Achse	Positionsfehler	Rundungsfehler
1	0.054 mrad	0.035 mrad
2	0.022 mrad	0.019 mrad
3	0.057 mrad	0.078 mrad
4	0.094 mrad	0.111 mrad
5	0.100 mrad	0.082 mrad
6	0.236 mrad	0.101 mrad
kartesisch	0.057 mm	0.063 mm
Bahnfehler	0.037 mm	nicht bestimmt

Tabelle 6.18: Genauigkeit des Manutec r2 Roboters nach Training der Bahnsteuerung von Bahn 1 mit 188 mm/s im Vergleich zu den Rundungsfehlern bei der Kommandierung des Roboters

Bahn	Geschw.	Positionsfehler ohne Eingriff	Positionsfehler mit Vorsteuerung	Positionsfehler mit Bahnsteuerung
1	375 mm/s	23.091 mm	0.245 mm	0.069 mm
1	188 mm/s	13.546 mm	0.174 mm	0.057 mm
2	375 mm/s	24.053 mm	0.210 mm	0.111 mm
2	188 mm/s	14.126 mm	0.161 mm	0.062 mm
3	500 mm/s	39.900 mm	0.209 mm	0.129 mm
3	250 mm/s	21.523 mm	0.157 mm	0.062 mm
4	375 mm/s	32.074 mm	0.183 mm	0.089 mm
5	375 mm/s	27.221 mm	0.206 mm	0.087 mm

Tabelle 6.19: Vergleich der kartesischen Positionsfehler bei Vorsteuerung bzw. Bahnsteuerung von verschiedenen Bahnen mit verschiedenen Geschwindigkeiten. Die Vorsteuerung wurde an Bahn 3 mit 500 mm/s trainiert.

von verschiedenen Bahnen zusammen. Als Faustformel wird durch die Bahnsteuerung sowohl der Positionsfehler als auch der Bahnfehler gegenüber der Vorsteuerung um den Faktor 2 verbessert. Es lohnt sich also bei wiederkehrenden Bewegungen, nach dem Abfahren einer Bahn mit Vorsteuerung noch eine Bahnsteuerung zu trainieren.

Die Tabellen 6.19 und 6.20 beschreiben das Ergebnis der jeweiligen Adaptionen, wobei bei der Bahnsteuerung auch das Modell an der zu korrigierenden Bahn identifiziert wurde. Im realen Einsatz, also bei Verbesserung einer Trajektorie, die mit adaptierter Vorsteuerung abgefahren wurde, stammt das Modell jedoch von der Trajektorie, an der die Vorsteuerung trainiert wurde. Meist wird keine neue Modellbildung erfolgen, da dazu eine Anregung nötig wäre. Das Training erfolgt dann also suboptimal.

Zum Vergleich wird eine Bahnsteuerung für Bahn 1 unter diesen Bedingungen trainiert, also mit einem Modell, das an Bahn 3 gebildet wurde. Damit wird der Positionsfehler von 0.069 mm nicht erreicht, sondern nur ein Wert von

Bahn	Geschw.	Bahnfehler ohne Eingriff	Bahnfehler mit Vorsteuerung	Bahnfehler mit Bahnsteuerung
1	375 mm/s	2.107 mm	0.125 mm	0.046 mm
1	188 mm/s	1.209 mm	0.093 mm	0.037 mm
2	375 mm/s	3.825 mm	0.125 mm	0.074 mm
2	188 mm/s	2.214 mm	0.101 mm	0.042 mm

Tabelle 6.20: Vergleich der kartesischen Bahnfehler bei Vorsteuerung bzw. Bahnsteuerung von verschiedenen Bahnen mit verschiedenen Geschwindigkeiten. Die Vorsteuerung wurde an Bahn 3 mit 500 mm/s trainiert.

0.115 mm. Hier sind also die Grenzen der Modellschätzung gefunden. Das Modell, das mit stehendem Greifer geschätzt wurde, kann die Bahn mit hängendem Greifer gegenüber der Vorsteuerung zwar verbessern, aber nicht so weit wie bei Verwendung des Modells für den richtigen Arbeitspunkt.

Daher wird das nachträgliche Training der Bahnsteuerung mit dem Modell und der Vorsteuerung aus Tabelle 6.5 versucht, also nach Training an stehendem und hängendem Greifer. Dabei wird die Genauigkeit aus Tabelle 6.19 erreicht. Es ist also sowohl zur reinen Vorsteuerung als auch zum weiteren Training sinnvoll, an mehr als einer Bahn zu trainieren.

6.5 Wirkung nichtlinearer Gütemaße

Gegenüber der Bahnsteuerung sind die Unterschiede durch Training der Vorsteuerung unter Verwendung eines anderen Gütekriteriums relativ klein. Tabelle 6.21 vergleicht die Kenngrößen bei Training mit den in Abschnitt 5.6.2 vorgestellten Varianten. Die Gütekriterien lassen sich sowohl zur Adaption einer Vorsteuerung als auch zum Training einer Bahnsteuerung verwenden.

Das Training geht wie in Abschnitt 6.1.3 von der an Bahn 3a trainierten all-

		Schmid'sche Bahn		Bahn 3a	
		rms Fehler	max. Fehler	rms Fehler	max. Fehler
1	Ohne Vorsteuerung	0.806 mm	5.369 mm	1.804 mm	2.451 mm
2	Allgemeine Vorsteuerung	0.180 mm	1.365 mm	0.176 mm	0.527 mm
	Adaptierte Vorsteuerung für				
3	minimalen rms Achsfehler	0.162 mm	0.808 mm	0.252 mm	0.729 mm
4	minimalen max. Achsfehler	0.163 mm	0.720 mm	0.312 mm	0.677 mm
5	minimalen rms Bahnfehler	0.140 mm	0.725 mm	0.293 mm	0.637 mm
6	minimalen max. Bahnfehler	0.140 mm	0.617 mm	0.290 mm	0.680 mm
	Bahnsteuerung für				
7	minimalen rms Achsfehler	0.066 mm	0.319 mm	-	-
8	minimalen max. Achsfehler	0.073 mm	0.241 mm	-	-
9	minimalen rms Bahnfehler	0.044 mm	0.184 mm	-	-
10	minimalen max. Bahnfehler	0.044 mm	0.120 mm	-	-

Tabelle 6.21: Quadratisch gemittelte und maximale Bahnabweichung bei unterschiedlichen Gütekriterien. Die allgemeine Vorsteuerung wurde an Bahn 3a trainiert. Alle anderen Vorsteuerungen wurden an der Schmid'schen Bahn adaptiert.

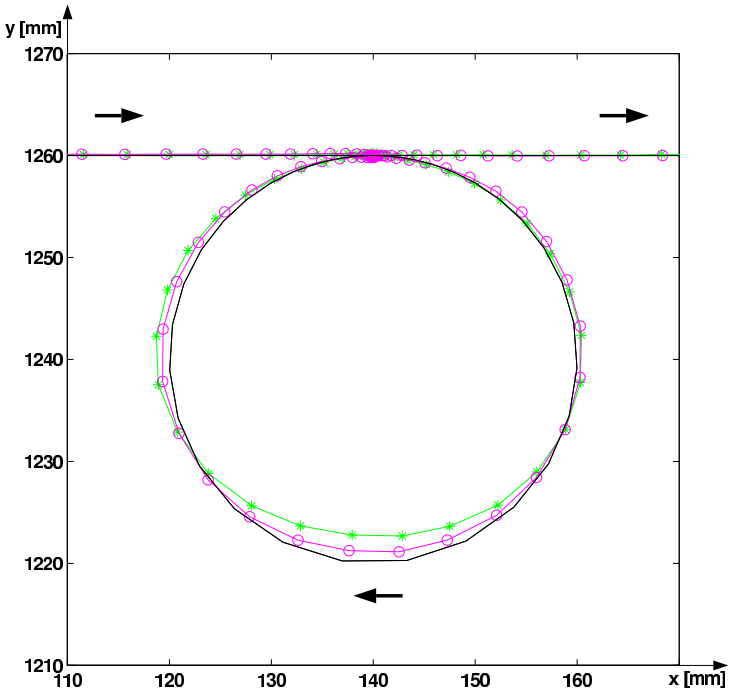


Abbildung 6.12: Kreis der Schmid'schen Bahn mit Vorsteuerung (∗ = allgemeine Vorsteuerung (Zeile 2), ○ = adaptierte Vorsteuerung für minimalen maximalen Bahnfehler (Zeile 6), schwarz durchgezogen = Soll-Bahn, Bahnfehler sind vergrößert dargestellt)

gemeinen Vorsteuerung (Zeile 2) aus.[2] Die folgenden Adaptionen erfolgen an der Schmid'schen Bahn und bauen jeweils aufeinander auf. Das bedeutet, dass ein Training mit neuem Gütekriterium erst begonnen wird, wenn in der jeweils vorangegangenen Adaption keine weitere Verbesserung erzielbar ist.

Die erste Anpassung der Vorsteuerung geschieht ohne nichtlineares Gütekriterium (Zeile 3). Sie ist erforderlich, um die an Bahn 3a trainierte allgemeine Vorsteuerung (Zeile 2) mit den verschiedenen nichtlinear trainierten Vorsteuerungen vergleichen zu können. Die Fortsetzung des Trainings von Bahn 3a an der Schmid'schen Bahn bringt dort leichte Verbesserungen, insbesondere bei den kritischen Bahnabschnitten.

Fortsetzung des Training unter spezieller Berücksichtigung des maximalen Fehlers (Zeile 4) bringt bei diesem eine weitere Verbesserung, aber auch die erwartete leichte Verschlechterung des Mittelwerts.

Eine direkte Minimierung des Bahnfehlers anstelle der Achsfehler (Zeile 5)

[2] Aufgrund eines anderen Endeffektors sind die kartesischen Bahnen verschoben, so dass sich andere Zahlwerte für die Bahnfehler egeben.

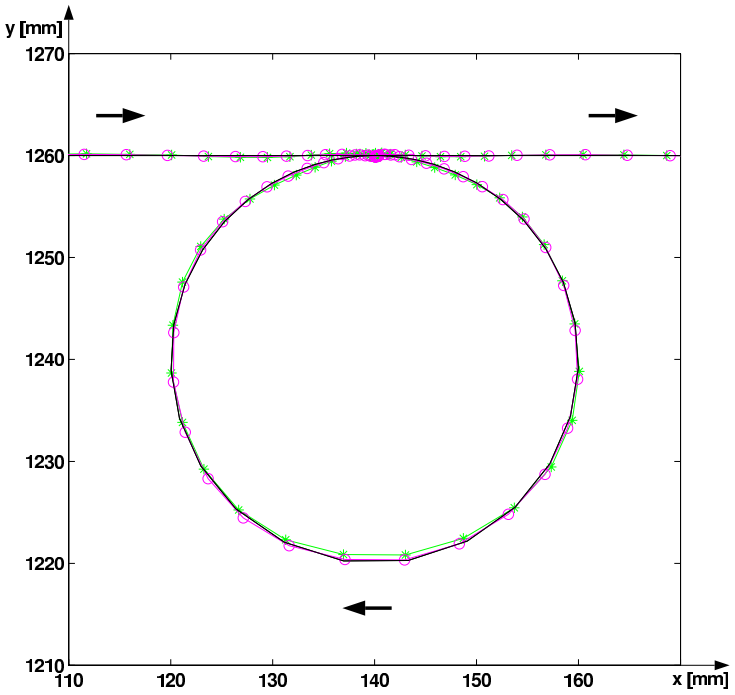


Abbildung 6.13: Kreis der Schmid'schen Bahn mit Bahnsteuerung für ($*$ = minimale rms Achsfehler (Zeile 7), $\circ$ = minimale maximale Bahnfehler (Zeile 10), schwarz durchgezogen = Soll-Bahn, Bahnfehler sind vergrößert dargestellt)

bringt, ausgehend von der linear adaptierten Vorsteuerung (Zeile 3), geringere Fehler bei beiden Kennwerten. Dafür verschlechtern sich die (nicht ausgewerteten) Orientierungs- und Geschwindigkeitsfehler.

Eine daran anschließende spezielle Reduktion der Maximalfehler (Zeile 6) bringt auch den gewünschten Erfolg. Dies ist in Abb. 6.12 im Vergleich zur allgemeinen Vorsteuerung demonstriert. In dieser Abbildung sind die Bahnabweichungen gegenüber der Soll-Bahn um den Faktor 2 vergrößert.

Insgesamt ist der Vorteil der nichtlinearen Gütekriterien bei der Vorsteuerung aber eher bescheiden. Bei der getesteten Bahn liegt er bei etwa 20 %.

Die Spezialisierung der Adaption auf die Schmid'sche Bahn bringt bei der ursprünglichen Trainingsbahn erwartungsgemäß eine Verschlechterung. Die immer noch gewährleistete Übertragbarkeit wird durch die nichtlinearen Gütekriterien kaum beeinträchtigt. In allen Fällen ist auch bei der zuletzt nicht trainierten Bahn eine wesentliche Verbesserung gegenüber dem Verhalten ohne Vorsteuerung (Zeile 1) messbar.

Die Wirkung der nichtlinearen Gütekriterien zur Bestimmung einer Bahnsteuerung ist qualitativ gleich. Quantitativ ergeben sich aber weit größere Unterschiede. So sinkt der maximale Bahnfehler gegenüber dem linearen Gütekriterium um 60 % (Abb. 6.13).

Auch in dieser Abbildung sind die Bahnabweichungen gegenüber der Soll-Bahn um den Faktor 2 vergrößert. Trotzdem ist im Gegensatz zur Vorsteuerung kaum ein Bahnfehler sichtbar. Das kommt daher, dass der lineare Ansatz der Vorsteuerungen nach Gleichungen wie (5.63) die dynamischen Effekte nicht mit ausreichender Genauigkeit beschreibt.

Bei den Kennwerten ist eine Größenordnung erreicht, die einerseits kaum nötig ist und die andererseits die Genauigkeit der externen Messung aus Abschnitt 6.1.3 übersteigt. Es kann davon ausgegangen werden, dass bei Training mit extern gemessenen Positionen alle Experimente mit Bahnsteuerung die Messgenauigkeit erreichen.

6.6 Anwendung der adaptiv vorausplanenden Steuerung auf sensorbasiert geplante Bahnen

Soweit beschreibt Kapitel 6 die erreichten Ergebnisse bei Anwendung der mittleren Ebene der in Abschnitt 4.2 vorgeschlagenen hierarchischen Architektur. In diesem Abschnitt werden die durch die obere Ebene entstehenden Eigenschaften diskutiert.

Dabei wird zunächst der prinzipielle Unterschied zu fest programmierten Bahnen vorgestellt. Dann folgen zwei Beispiele von sensorgestützten Regelungen.

6.6.1 Prinzipielles Problem

Problematisch bei sensorgestützten Regelungen nach der multisensoriellen Architektur aus Abschnitt 4.2 ist insbesondere, dass die Schnittstelle zum *idealen Roboter* neben den aktuellen Sensorwerten auch die zukünftige Entwicklung enthält. Diese Daten können die in Abschnitt 4.4 vorgestellten Sensorsysteme nicht immer bereitstellen.

Während beim Bildverarbeitungssystem die Prädiktion durch eine geschickte

Auswertung des Bildinhalts möglich ist, erfordern Prädiktionen bei anderen Sensoren einen physikalisch anderen Aufbau oder Erweiterungen. Bei solchen Sensoren bleibt daher nur die Extrapolation der bisher ermittelten Bahn.

Bei der Fusion von programmierten Bahndaten (Referenzbahn) und Sensordaten sind prädiktive Bahninformationen verfügbar. Die Integration von extrapolierten Sensordaten ist je nach Aufgabenstellung unter Umständen problematisch.

6.6.2 Kraftregelung

In Abschnitt 3.3.1 wurde die Aufgabe einer kraftgeregelten Konturverfolgung durch adaptive direkte Regelung beschrieben. Hier folgt die Bearbeitung der Aufgabe durch die multisensorielle Struktur nach Abb. 4.15. Dabei ist die Erläuterung der violett dargestellten Funktionsblöcke Sensordatenverarbeitung und Sensorintegration der Übersichtlichkeit halber in den Anhang D.1 verschoben.

Der Sensor liefert keine prädiktiven Messwerte. Zunächst erfolgt auch keine Prädiktion der Kraft. Die Bahnplanung liefert in jedem Abtastschritt also Soll-Positionen, die sich nur senkrecht zur aktuellen Kraftrichtung um die in Positionsinkremente umgerechnete Soll-Geschwindigkeit unterscheiden. In Kraftrichtung liegt diese Soll-Bahn so, dass an der aktuellen Position der (betragsmäßig gegebene) Kraftsollwert eingehalten wird. Kraftänderungen können nicht vorhergesagt werden.

Aufgrund von Gleichung (5.53) ergibt sich, dass unter diesen Umständen die Vorsteuerung nur auf die Bewegung tangential zur Kontur wirkt, da alle Differenzen $x_{dn}(k+i) - x_{dn}(k)$ und $\tau_d^c(k+i) - \tau_d^c$ null sind. Dabei ist x_{dn} die Komponente der kartesischen Soll-Position $\mathbf{x}_d$, die normal zur Kontur steht.

Bei Bewegungen mit Kraftkontakt ist es sinnvoll, neben einem Positionsfehler auch einen Kraftfehler, also die Abweichung des Betrages des gemessenen Kraftvektors von der Soll-Kraft $f_d = 10$ N zu betrachten. Für die gesamte Aufgabe der Konturverfolgung ist so ein Kraftfehler aussagefähiger als ein Bahnfehler, da er auch die Qualität der oberen Ebene der hierarchischen Struktur bewertet.

Zunächst wird die Kontur (siehe Abb. 3.17) bei einer Geschwindigkeit von 38 mm/s ohne Vorsteuerung ertastet. Dabei beträgt der mittlere Kraftfehler

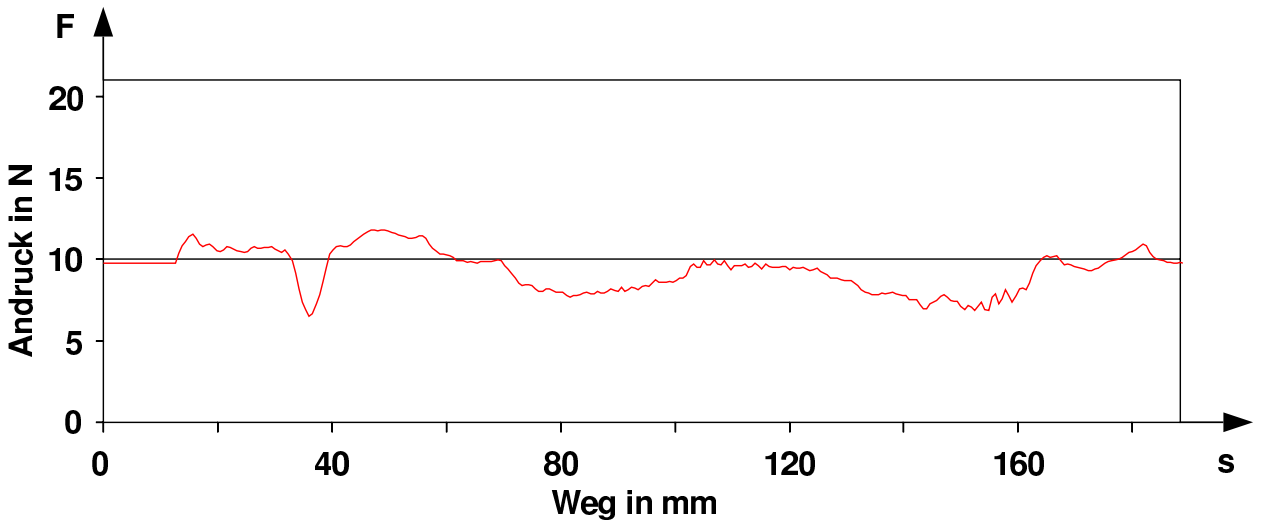


Abbildung 6.14: Kraftfehler beim Ertasten der Kontur ohne Vorsteuerung bei 38 mm/s

etwa 1.2 N (siehe Abb. 6.14). Am Verlauf des Kraftfehlers kann man die Ecken und Krümmungen der Kontur erkennen. Die Ecke wird nach 35 mm erreicht, der folgende konkave Bereich geht bis etwa 70 mm, der konkave Bereich am Ende beginnt bei etwa 160 mm.

Qualitativ unterscheiden sich die Kraftverläufe von Abb. 6.14 und 3.19 wenig. Dabei wurde Abb. 3.19 bei direkter Rückführung der Kraftwerte[3] ohne Berechnung der jeweiligen Soll-Positionen, aufgezeichnet. Quantitativ ist die direkte Regelung sogar besser (mittlerer Kraftfehler von 0.7 N anstelle von 1.2 N), da die Adaption des Reglers dort implizit eine Extrapolation der Sensorwerte vornimmt. Dadurch wird die mittlere Krümmung der Kontur vorgesteuert (ähnlich wie in [173]).

Aus Kausalitätsgründen kann die Bahn während des Tastens selbst nicht genau abgefahren werden, da in jedem Abtastschritt der Soll-Bahnpunkt erst durch die Kraftmessung definiert wird. Zusätzlich zu den in Abschnitt 3.2.1 und 3.2.2 auftretenden Bahnfehlern gibt es also noch einen weiteren Fehler. Aufgrund der Signalverarbeitungstotzeiten und des Schleppfehlers ergibt sich eine maximale Geschwindigkeit, bei der der Messbereich des Sensors gerade noch nicht überschritten wird. Diese Geschwindigkeit liegt bei der Kontur aus Abb. 3.17 weit unter den bei freien Bewegungen getesteten Werten.

[3] Auch der hier als direkte Kraftregelung bezeichnete Ansatz verwendet die Positionsschnittstelle des Roboters mit der Kaskadenregelung, nicht aber die mittlere Ebene mit der Vorsteuerung zur Realisierung eines *idealen Roboters*.

	Positionsfehler	Kraftfehler
ohne Vorsteuerung	2.920 mm	1.18 N
mit linearer Vorsteuerung von Bahn 3	0.445 mm	0.79 N
mit aufgabenspezifischer Vorsteuerung für Kontur	0.397 mm	0.67 N
bei direkter aufgabenspezifischer Kraftregelung	-	0.7 N

Tabelle 6.22: Mittlere Fehler beim Ertasten der Kontur mit 0.6 mm/Schritt

Beim Ertasten der Kontur mit dem an Bahn 3 bestimmten linearen Vorfilter wird ein mittlerer Kraftfehler bezüglich der x-y-Ebene von etwa 0.8 N eingehalten. Er ist ein Maß für den Bahnfehler, also die Komponente des Positionsfehlers senkrecht zur Bewegungsrichtung. Im Vergleich zum Ertasten der Bahn ohne Vorsteuerung verbessert sich diese Komponente nur geringfügig (siehe Tabelle 6.22). Das kommt daher, dass die prädizierten Bahnpunkte nach Gleichung (D.6), abgesehen vom Vorschub, nur durch den zuletzt vermessenen Kontaktpunkt vorhergesagt werden, was bei Krümmungen oder ungenau gemessenen Richtungsvektoren sicherlich nicht der richtigen Soll-Bahn entspricht. Eine an bekannten Bahnen trainierte Vorsteuerung korrigiert nur dann richtig, wenn die prädizierten Soll-Positionen sich nachträglich bestätigen.

Zur Verringerung des Kraftfehlers während des Ertastens der Bahn gibt es, abgesehen von der Reduktion der Bahngeschwindigkeit, zwei Möglichkeiten: Entweder man verbessert die Prädiktion der Kontur oder man ändert die Regelung. Dabei steht die zweite Möglichkeit im Widerspruch zur hierarchischen Struktur aus Abschnitt 4.2, bei der Regelung und Aufgabe voneinander unabhängig sind.

Als Änderung der Regelung ist insbesondere die Einführung einer Messwertrückführung geeignet, da bei Krümmungen der Kontur im Gegensatz zu den Voraussetzungen im Abschnitt 5.2.1 durch die Vorsteuerung nicht die Einhaltung der Soll-Bahn garantiert wird. Während Abschnitt 5.2.1 davon ausgeht, dass die Vorsteuerung den Endeffektor von einem Punkt auf der Soll-Bahn zum nächsten Punkt der Soll-Bahn führen soll, berücksichtigt die Messwertrückführung nach Abschnitt 5.2.2, dass der Ausgangspunkt ggf. nicht auf der Soll-Bahn liegt, weil er falsch prädiziert wurde. Die Messwertrückführung regelt damit bei sensorgestützter Bahnplanung nicht nur Störungen aus, sondern auch Prädiktionsfehler.

Es zeigt sich aber, dass der Regler aus Abschnitt 5.4.2 beim Ertasten einer Kontur sehr schnell die Beschleunigungsgrenzen der Robotersteuerung erreicht. Da die Gewichtsfunktionen nach Abb. 5.2 die Maximalwerte erst im zweiten

Schritt nach der Totzeit erreichen, kann eine Regeldifferenz zum Zeitpunkt k eine Stellgröße bewirken, die bis zum Zeitpunkt $k + n_t + 1$, abgesehen von Krümmungen, auf die Soll-Bahn führt, sie aber im Schritt $k + n_t + 2$ schon wieder verlässt, da die nächste Stellgröße, das Abbremsen, nicht vollständig ausgeführt werden kann.[4]

Als Alternative kann ein Regler speziell für den bisher nicht berücksichtigten Fall von Stellgrößenbegrenzungen trainiert werden. Dieser Regler korrigiert die Regeldifferenzen mit kleineren Stellgrößen, also verlangsamt. Das entspricht der Wahl einer suboptimalen Referenztrajektorie wie bei Geyer und Rake [91]. Die Unterschiede beim Training gegenüber Kapitel 5 betreffen besonders die Stellgrößenbestimmung. Dabei muss ein hoher Wert für die angenommene Störung σ^2 vorgegeben werden. Dies entspricht einer starken Gewichtung der Stellgrößen, wie sie schon in Abschnitt 5.2.1 als Alternative zur Minimierung eines quadratischen Gütekriteriums nach Gleichung (5.30) vorgeschlagen wurde.

Experimente mit verschiedenen Realisierungen von Messwertrückführungen zeigen beim Manutec r2 Roboter keine signifikante Verbesserung gegenüber der reinen Vorsteuerung. Dies liegt daran, dass der Roboter durch die Vorsteuerung, sofern keine weiteren unvorhergesehen Einflüsse vorliegen, innerhalb weniger Schritte die Soll-Bahn erreicht. Aufgrund von Totzeit und Stellgrößenbegrenzungen kann diese Zeit selbst durch einen idealen Regler kaum unterboten werden.

Eine Verbesserung des Verhaltens ist daher nur durch Prädiktion möglich. Dabei kann sowohl die zukünftige Bahn vorhergesagt werden als auch die künftigen Kontaktkräfte, wobei Letzteres sicherlich schwieriger ist, da die Kraftänderungen aus den Vorhersagefehlern der Bahn entstehen. Daher wird nur die Soll-Bahn prädiziert.

Dabei können die zukünftigen Achs-Sollwerte entweder explizit oder implizit aus den bisherigen geschätzt werden.

Bei expliziter Prädiktion werden die Parameter k_i aus

$$\hat{w}(k+i) = w(k) + \sum_{i=1}^{n_p} k_i \cdot (w(k-i) - w(k)) \tag{6.1}$$

[4]Das beschriebene Szenario ist hier kein Indiz für einen instabilen Reglerentwurf, sondern nur für eine nicht berücksichtigte Stellgrößenbegrenzung.

geschätzt und die Ergebnisse zur Vorsteuerung nach Gleichung (5.44) genutzt, während bei impliziter Prädiktion die Parameter r_{pi} der Vorsteuerung

$$u(k) = w(k) + \sum_{i=1}^{n_w} r_{wi} \cdot (\hat{w}(k+i+n_t) - w(k)) + \sum_{i=1}^{n_p} r_{pi} \cdot (w(k-i) - w(k)) \quad (6.2)$$

direkt aufgabenspezifisch trainiert werden. Dabei wird $\hat{w}$ unter der Annahme eines konstanten Kraftrichtungsvektors als $\hat{\mathbf{x}}_d.$ nach Gleichung (D.6) bestimmt.

Anstelle der einfachen expliziten Schätzung sind auch verfeinerte Methoden, z. B. nach [68] möglich. Trotzdem wird hier die implizite Prädiktion der Kontur verwendet, da durch die direkte Adaption auch eine Filterung stattfindet. Bei expliziter Schätzung geht dagegen die Zuverlässigkeit der Vorhersagen nicht in das Training der Vorsteuerung ein. Voraussetzung der Anwendbarkeit der impliziten Schätzung ist allerdings, dass beim Training der Optimierung nach Abschnitt 5.2.1 die richtigen Regeldifferenzen

$$e(k+i) = w(k+i) - y(k+i) \quad (6.3)$$

vorliegen. Die $\hat{w}$ entsprechend Gleichung (D.6) dienen also nur zur Vorfilterbestimmung.

Training eines solchen aufgabenabhängigen Vorfilters an der Kontur ergibt mit $n_p = 5$ die dritte Zeile von Tabelle 6.22, also auch nur eine geringe Verbesserung. Das bestätigt die Aussage, dass die Konturverfolgungsaufgabe in dieser Anordnung nicht exakt lösbar ist. Aufgrund der Stellgrößenbegrenzungen führen Vorsteuerung, Messwertrückführung und aufgabenspezifische Vorsteuerung zu etwa dem gleichen Ergebnis.

Ein anderer Grund für die begrenzte Güte liegt in der endlichen Auflösung der Roboterposition (vergl. Abb. 6.7), wodurch nach [97] ohne weitere Einflüsse eine Dauerschwingung entsteht.

Die aufgabenabhängige Vorsteuerung erreicht also auch nur die Güte der direkten Kraftregelung aus Abschnitt 3.3.1. Entgegen einer expliziten Prädiktion verletzt aber auch diese Lösung die Unabhängigkeit von Regelung und Aufgabenstellung. Hier zeigt sich, dass die allgemeine hierarchische Struktur aus

	Geschwindigkeit	Positionsfehler	Kraftfehler
Ertasten der Kontur	0.6 mm/Schritt	0.433 mm	0.80 N
Wiederholen der Bahn	0.6 mm/Schritt	0.187 mm	0.24 N
Wiederholen der Bahn	1.2 mm/Schritt	0.305 mm	0.41 N

Tabelle 6.23: Mittlere Fehler der ertasteten Bahn bei linearer Vorsteuerung

Abschnitt 4.2 einer aufgabenabhängig entworfenen speziellen Struktur unterlegen ist.

Beim nochmaligen Abfahren der Bahn kann man dann die Kenntnis der Kontur voraussetzen und die gespeicherte Bahn ohne weitere Sensorauswertung abfahren. Das ist dann das Abfahren einer offline bestimmten Bahn, bei der eine Messwertrückführung nach den Ergebnissen von Abschnitt 6.1 nicht nötig ist. Eine Vorsteuerung empfiehlt sich allerdings, da sonst keine Verbesserung gegenüber dem Ertasten möglich ist.

Durch die lineare Vorsteuerung, die an Bahn 3 trainiert wurde, reduziert sich der mittlere Kraftfehler beim Wiederholen der Bahn deutlich (siehe Tabelle 6.23). Insbesondere verschwindet der Einbruch nach 30 mm (Abb. 6.15) an der Ecke der Kontur in Abb. 3.17, der beim Ertasten nicht vorhersehbar war. Ebenso werden die zu hohen Kräfte im Bereich der beiden konkaven Bereiche bei einer Bahnwiederholung ausgeglichen.

Der Kraftfehler entspricht dem tatsächlichen Bahnfehler entsprechend Ab-

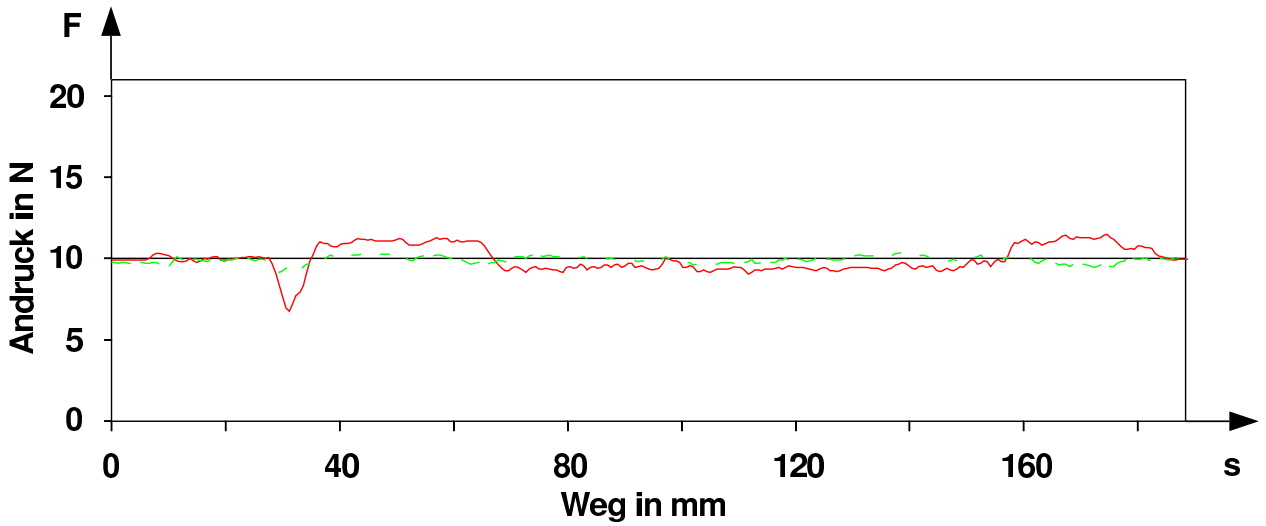


Abbildung 6.15: Kraftverlauf beim wiederholten Abfahren der ertasteten Bahn mit linearer Vorsteuerung (rot durchgezogen = Ertasten, grün gestrichelt = Wiederholen

schnitt 3.4. Er enthält die Summe aus der Ungenauigkeit bei der Bestimmung der Kontur und dem gesamten Fehler bei der Bahnwiederholung. Mit einem Kraftfehler von etwa 0.2 N entsprechend einem Bahnfehler von 0.1 mm kann man also zufrieden sein.

Die bei Wiederholung der ertasteten Bahn auftretenden Fehler lassen auch die Möglichkeiten eines prädiktiven Sensors beim Ertasten erkennen. Bei gleicher Geschwindigkeit ist also eine Verbesserung um den Faktor 3 möglich (0.24 N anstelle von 0.80 N), wenn der Sensor die Bahn über die gesamte Vorsteuerzeit von $n_t + n_w = 10$ Schritten vorhersagen kann. Bei gleichem Kraftfehler ist eine Erhöhung der Geschwindigkeit möglich (siehe auch [202]).

Bei Bahnwiederholung mit doppelter Geschwindigkeit ergibt sich wieder eine leichte Verschlechterung. Der Kraftfehler ist aber auch dort noch deutlich geringer als ohne Vorsteuerung oder direkt beim Ertasten. Bei höheren Geschwindigkeiten wird dann eine Kraftvorsteuerung nötig (siehe Tabelle 6.15).

6.6.3 Tracking mit visuellem Sensor

In diesem Abschnitt wird die Aufgabe aus Abschnitt 3.3.2 betrachtet, also die Verfolgung einer Linie, die visuell erfasst wird (siehe Abb. 6.16 und 6.18). Bei diesem Experiment wird von einer (quasi-)statischen Umgebung ausgegangen, bei der eine Kamera zur Erfassung des aktuellen Zustands dient. Da dieser Zustand sich nicht nur auf die Werkzeugspitze des Roboters bezieht, wird auch Information erfasst, die bei anderen Systemen durch einen prädiktiven Sensor gemessen wird. Ein Bildverarbeitungssystem kann also als 2D Sensor sowohl die Messung der aktuellen Abweichung von der Soll-Bahn als auch die Vorhersage der zukünftigen Regeldifferenzen aufgrund von Krümmungen der sensierten Linie übernehmen.

In Erweiterung zu Abschnitt 3.3.2 wird keine gerade Kante, sondern eine geschwungene Linie abgefahren, wobei die Linie durch ein Kabel realisiert ist. Die gesamte Szene wird durch 4 Punkte der Linie repräsentiert (Abb. 6.17), die nach Anhang D.2 aufgenommen werden (siehe auch Abb. 6.19). Dabei beschreibt die Referenzbahn eine gerade Linie, die vom Endeffektor mit maximaler Beschleunigung und Verzögerung abgefahren wird. In der Mitte der Bahn wird eine Geschwindigkeit von 1.25 m/s erreicht. Die Kamera ist im Abstand von 35 cm von der Linie am Endeffektor montiert. Daraus ergibt sich aufgrund der Ka-

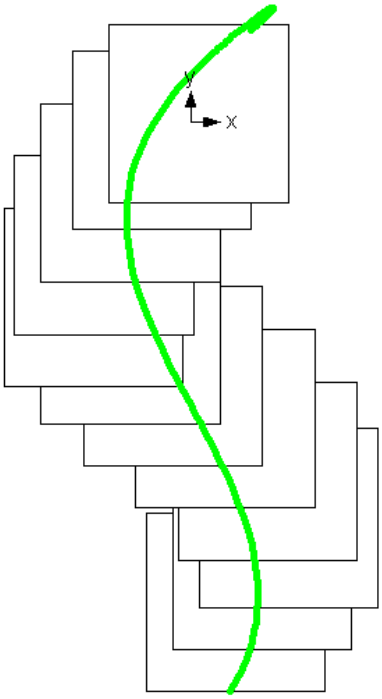


Abbildung 6.16: Laufende Aufnahme von Bildern beim Verfolgung einer gekrümmten Linie

merageometrie ein vertikal sichtbarer Bereich von ±17 cm und eine horizontale Auflösung von 0.6 mm/Pixel.

Die Qualität der Regelung hängt davon ab, wie stark die Linie gekrümmt ist. Bei geringer Krümmung wird eine mittlere Abweichung der im Bild gekennzeichneten Linie vom TCP von 0.3 mm erreicht (Abb. 6.20). Verbleibende Fehler kommen von der Ungenauigkeit der Vorsteuerung und der Bildverarbeitung.

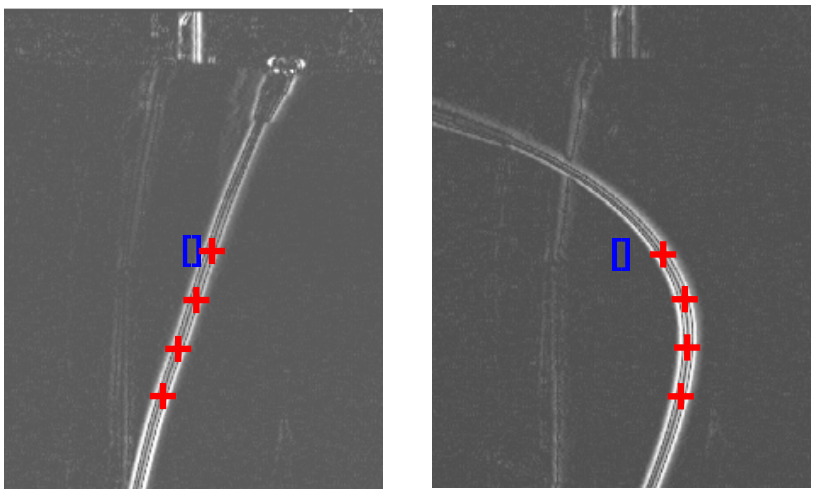


Abbildung 6.17: Gefilterte Bilder des Anfangs von zwei Beispielbahnen ([] = TCP; + = erfasste Punkte der Linie)

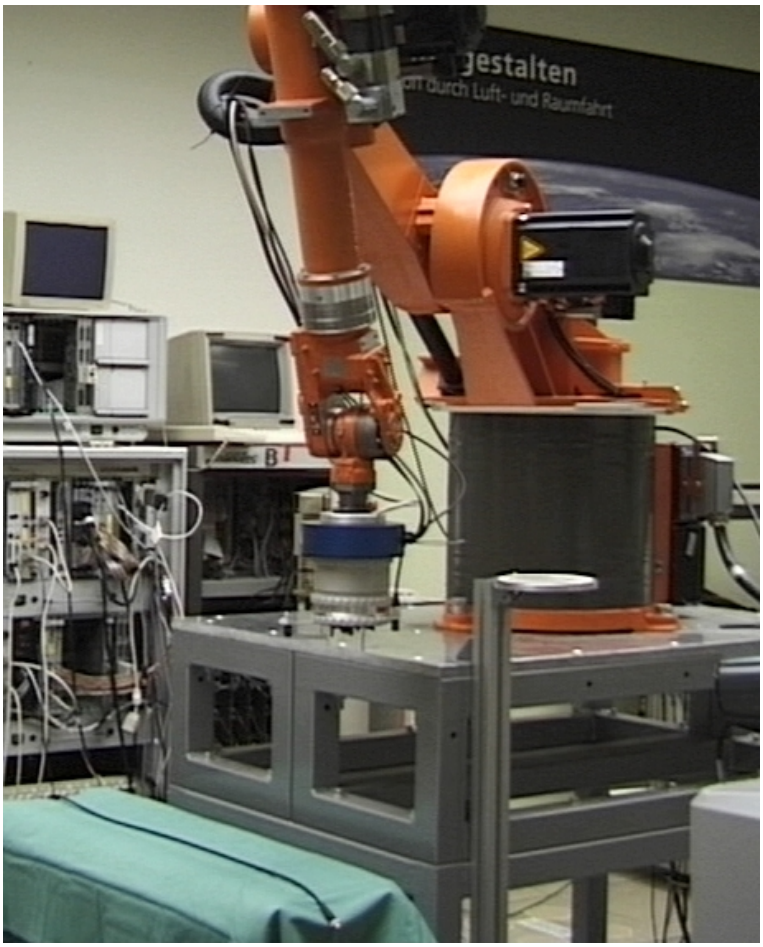


Abbildung 6.18: Anordnung zur Verfolgung einer gekrümmten Linie

Insbesondere wirken sich Nachgiebigkeiten am Anfang und am Ende der Bahn aus.

Zum Vergleich enthält Abb. 6.20 auch das Experiment mit der direkten Regelung (vergl. Abschnitt 3.3.2) durch einen adaptierten PD-Regler. Das Verhalten ist mit einer mittleren Abweichung von etwa 5 mm deutlich schlechter. Dies ist nicht verwunderlich, da nur der dem TCP nächstliegende Punkt der Linie ausgewertet wird. Dadurch ergibt sich am Anfang der bei rampenförmigen Sollwertänderungen typische Schleppfehler, am Ende der Bahn ist ein Überschwingen sichtbar, da der PD-Regler implizit die Orientierung der Linie im Raum extrapoliert. Ein weiterer Grund für das schlechtere Verhalten der direkten Regelung liegt darin, dass die dynamischen Eigenschaften des Roboters nicht berücksichtigt werden. Das spielt in diesem Fall aber eine untergeordnete Rolle.

Bei stärkerer Krümmung der Linie treten Bewegungsunschärfen in den Bildern

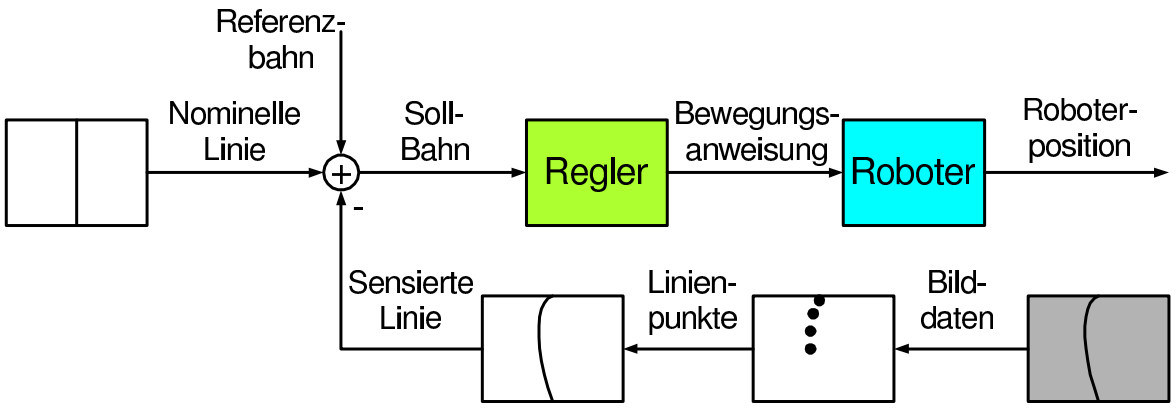


Abbildung 6.19: Extraktion der Ist-Bahn aus einzelnen Linienpunkten und Regelung des Roboters so, dass die nominellen Sensordaten erreicht werden

auf. Dadurch wird die Bahngenauigkeit mit Prädiktion etwa um den Faktor zwei verschlechtert. Dagegen erlauben die neuen Versuchsbedingungen bei der nicht prädiktiven direkten Regelung keine Verfolgung der Linie (Abb. 6.21 und 6.22).

Noch höhere Krümmungen der Bahn, bzw. noch größere Unterschiede zwischen der Orientierung der Referenzbahn und der Linie können von keinem der betrachteten Systeme beherrscht werden. Das liegt daran, dass die Referenzbahn mit den maximal erlaubten Beschleunigungen des Roboters entworfen wurde. Merkliche Bewegungen senkrecht dazu führen zum Verlassen des Raumes der erlaubten Betriebszustände. Dadurch werden in den Getrieben Schwingungen

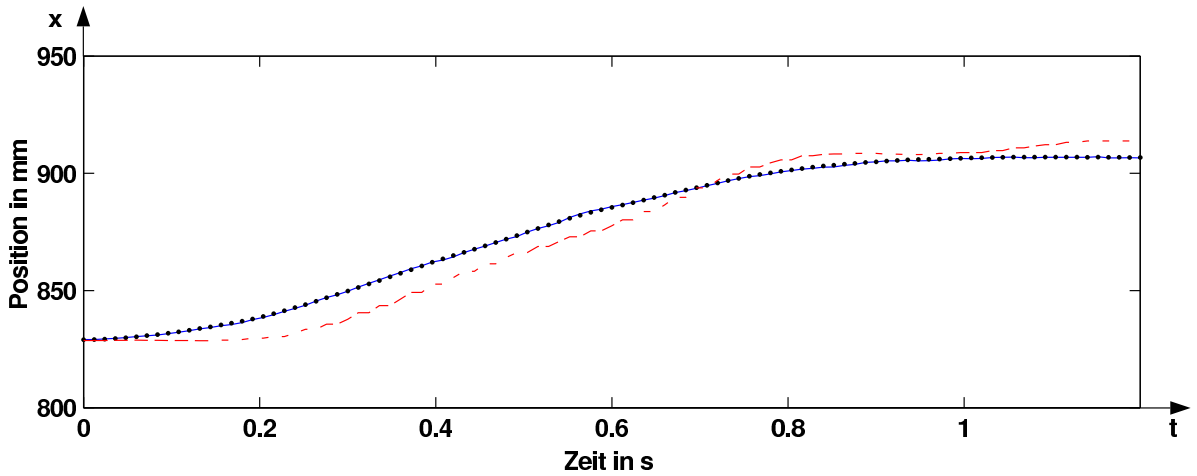


Abbildung 6.20: Bewegung bei geringer Krümmung der Linie (schwarz gepunktet = Verhalten mit prädiktiver Vorsteuerung; rot gestrichelt = Verhalten mit direkter PD-Regelung; blau durchgezogen = sensorisch vermessene Linie)

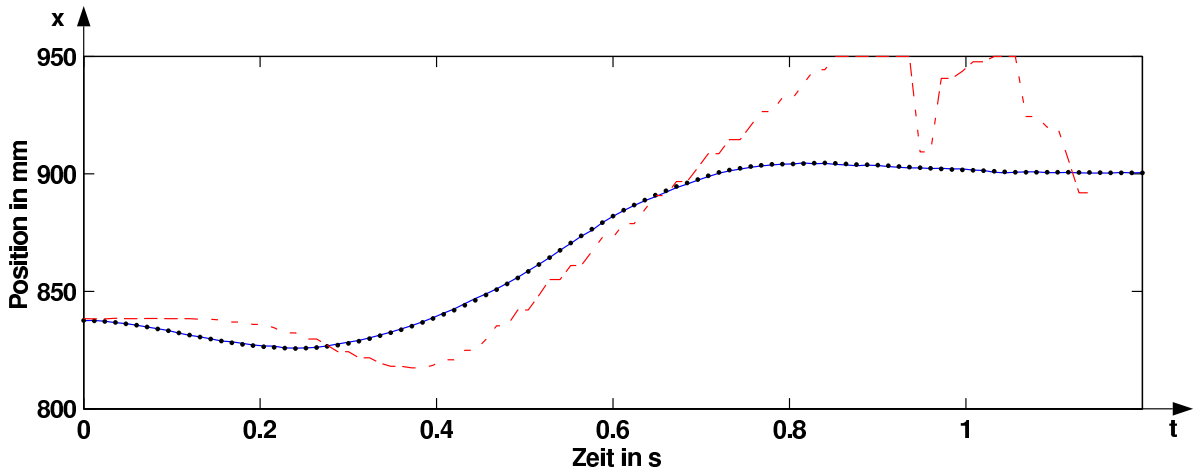


Abbildung 6.21: Bewegung bei starker Krümmung der Linie (schwarz gepunktet = Verhalten mit prädiktiver Vorsteuerung; rot gestrichelt = Verhalten mit direkter PD-Regelung; blau durchgezogen = sensorisch vermessene Linie)

angeregt, die eine gesonderte Behandlung erfordern, was über den Umfang dieser Arbeit hinausgeht.

Interessanterweise ist das Verhalten annähernd unabhängig davon, ob eine Vorsteuerung der durch die Bildverarbeitung sensorisch erfassten Soll-Positionen erfolgt. Das liegt an der bei dieser Bahn sehr hohen Bahngenauigkeit, die ohne Vorsteuerung schon die durch die Bildverarbeitung gesetzten Grenzen erreicht. Der prädiktive Charakter der Sensordatenverarbeitung im vorgeschlagenen dreistufigen System wird ohne Vorsteuerung dadurch erreicht, dass die sensorisch erfasste Position der Linie jeweils der Soll-Position entspricht, die gegenüber der

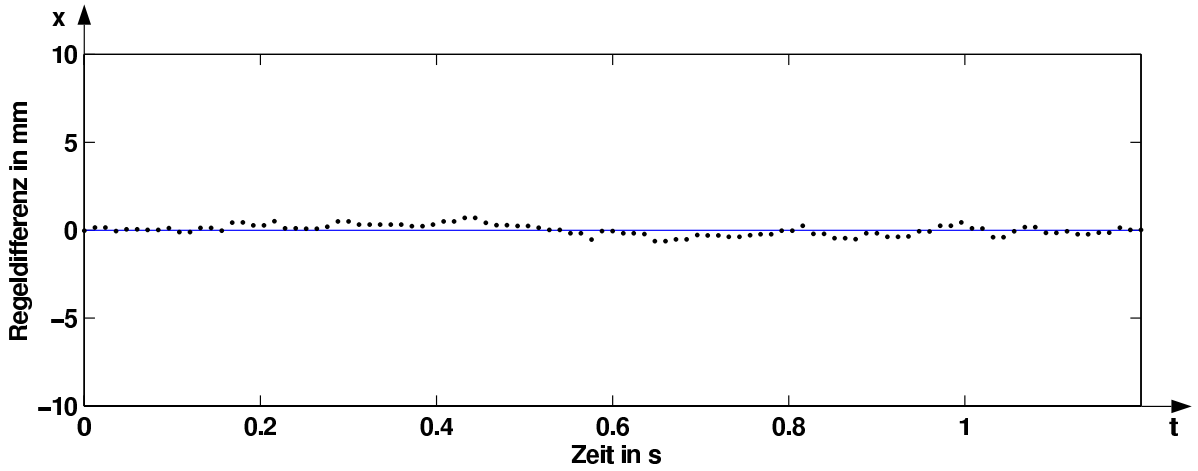


Abbildung 6.22: Regeldifferenz beim Experiment nach Abb. 6.21 mit prädiktiver Vorsteuerung (Sensorisch vermessener Abstand der Linie vom TCP)

aktuellen Roboterposition um den Schleppfehler vorauseilt. So heben sich die Wirkungen der Verzögerung der Regelung und der nicht vorhandenen Prädiktion weitgehend auf.

Unabhängig von der Vorsteuerung muss der Sichtbereich der Kamera dem Bereich entsprechen, der innerhalb der nächsten n_w Abtastschritte abgefahren wird. Der Teil der Trajektorie, der in der dreistufigen Architektur an den *idealen Roboter* weitergegeben wird, muss also verfügbar sein. Dazu kommt wegen der langsamen Sensorik noch eine Reserve, sodass immer n_w Abtastschritte vorhergesagt werden können. Bei der im Experiment erreichten Spitzengeschwindigkeit von 15 mm/Schritt sind das etwa 25 cm. Der sichtbare Bereich der Kamera von 17 cm reicht also nicht. Dadurch können starke Krümmungen im Bereich der maximalen Geschwindigkeit (Bahnmitte) nicht fehlerfrei nachgefahren werden.

Es gibt zwei Alternativen: Entweder man wählt eine Kamera mit kürzerer Brennweite und damit größerem Blickwinkel, oder man schwenkt die Kamera gegenüber der Bewegungsrichtung. In beiden Fällen wird ein größerer Bereich der zukünftigen Trajektorie sichtbar, allerdings sinkt auch die Auflösung, zumindest in größerer Entfernung vom TCP. Beim Schwenken der Kamera muss außerdem die Orientierung der aktuellen Linienrichtung nachgeführt werden, was ggf. nicht schnell genug möglich ist. Daher wird auf eine Vergrößerung des Blickfeldes verzichtet, auch wenn dadurch in manchen Fällen eine höhere Genauigkeit erreicht werden könnte.

Dieser Abschnitt zeigt, dass die hohe Bahngenauigkeit des vorgesteuerten Roboters durch das multisensorielle Konzept auch bei sehr schnellen sensorgeführten Bewegungen annähernd errreicht wird. Damit wird ein Vorurteil widerlegt, dass Bildverarbeitungssysteme langsam und unpräzise sind und daher bei schnellen Bewegungen nicht zur Sensorrückkopplung eingesetzt werden können.

6.7 Diskussion und Bewertung der erzielten Ergebnisse

Die Bahnfehler lassen sich, soweit sie durch die roboterinternen Messsysteme erfassbar sind, fast vollständig reduzieren (siehe Tabelle 6.24). Wie erwartet, konnte bei schnellen Bahnen allerdings die Positionier(wiederhol)genauigkeit des Roboters nicht erreicht werden.

Messmethode	erreichbarer Wert	
intern gemessener mittl. Achspositionsfehler	ca. 0.5 mrad	ca. 1 %
intern gemessener mittl. kart. Bahnfehler	ca. 0.3 mm	ca. 10 %
extern gemessener mittl. kart. Bahnfehler (nur KUKA)	ca. 1.0 mm	ca. 30 %

Tabelle 6.24: Durch lineare Vorsteuerung erreichbare mittlere Achs- bzw. Bahnfehler bei den untersuchten Robotern (Relativwerte bezogen auf die Abweichungen der Ist-Positionen von den Soll-Positionen bei ausschließlicher Verwendung der industriellen Steuerung)

Erweiterungen der Methode zur Berücksichtigung von Kopplungen zwischen den Achsen oder von externen Kräften konnten am Experiment verifiziert werden (siehe Tabelle 6.25). Es konnte aber auch gezeigt werden, dass mindestens dasselbe Ergebnis, nämlich eine Halbierung des Restfehlers, einfacher erreicht werden kann, wenn die auszuführende Trajektorie schon während der Adaptionsphase bekannt ist und damit eine Bahnsteuerung erlaubt. Die Wahl spezieller Gütekriterien führt bei den entsprechenden Größen zu einen weiteren leichten Verbesserung.

Methode	Reduktion von	um ca.
Bahnsteuerung für feste Trajektorie	mittl. Achs- und Bahnfehler	50 %
Neuronales Netz	mittl. Achspositionsfehler	50 %
Kraftvorsteuerung	mittl. kart. Bahnfehler	40 %
explizite Minimierung der Maximalwerte	max. Achs- und Bahnfehler	20 %
explizite Minimierung der kart. Bahnfehler	mittl. und max. kart. Bahnfehler	20 %

Tabelle 6.25: Bei interner Messung zusätzlich erreichbare Reduktion der Achs- bzw. Bahnfehler, bezogen auf die Fehler nach Tabelle 6.24 ohne Einsatz der entsprechenden Methode (Neuronale Netze und Kraftvorsteuerung sind nicht in Kombination mit der Bahnsteuerung anwendbar, liefern also keine weitere Verbesserung)

Die Adaption konvergiert bei allen dokumentierten Reglerelementen robust innerhalb weniger Trajektorien, wobei die jeweils benötigte Rechenzeit selbst beim Einsatz Neuronaler Netze kaum eine Rolle spielt. Ein Einsatz von zusätzlichen Signalprozessoren ist also nicht erforderlich.

Obwohl die adaptierten Vorsteuerungen theoretisch nur für den Arbeitsbereich optimal sind, in dem die Parameter angepasst wurden, zeigt sich, dass auch in grundlegend anderen Konfigurationen Verbesserungen erzielt werden. Durch die hoch übersetzenden Getriebe und den Ansatz im Achsraum machen dabei auch die Nähe zu Singularitäten oder extreme Trägheitsmomente kaum etwas aus.[5]

[5]Grundsätzlich ist die kartesische Genauigkeit an den Grenzen des Arbeitsbereichs bei gleichen Gelenkfehlern sicherlich schlechter, da die Hebelarme größer sind.

Lediglich die nichtlinearen Anteile der Vorsteuerung sind von den Trainingskonfigurationen abhängig. Die Allgemeingültigkeit der Berücksichtigung von Kontaktkräften konnte nicht getestet werden. Sie ist sicherlich auch nur bei Training in unterschiedlichen Konfigurationen gegeben. Der Einfluss dieser beiden Elemente der Vorsteuerung ist aber grundsätzlich gering.

Es ist zu erwarten, dass die Adaptionen bei Robotern von anderen Herstellern in gleicher Weise konvergieren. Bezüglich der Konfiguration kann es dabei leichte Unterschiede geben. So erfordern ggf. andere Kopplungen einen anderen Einsatz von Neuronalen Netzen.

Bei den durchgeführten Experimenten wurden auch Einschränkungen sichtbar, die zu Forderungen an die nächste Generation von Robotersteuerungen führen.

Dies ist zunächst das Problem der Messbarkeit der tatsächlichen Position des Endeffektors. Neben den in dieser Arbeit nicht betrachteten statischen Positionsfehlern müssen auch die dynamischen Nachgiebigkeiten erfasst werden, wenn ohne komplizierte externe Vermessung im normalen Betrieb mit hoher Genauigkeit gefahren werden soll. In dieser Arbeit wurde gezeigt, dass zumindest bei den (schnelleren) Robotern der Firma Kuka Gelenkelastizitäten existieren, die im Idealfall für jeden Roboter und jede Bahn neu vermessen werden müssen. Dabei war die im Experiment verfügbare Messgenauigkeit der einschränkende Faktor für die Verbesserung der Bahngenauigkeit (siehe Tabelle 6.24). In der Praxis müssen solche dynamischen Kalibrierungen aufgrund von Erwärmung oder Abnutzung sicherlich von Zeit zu Zeit wiederholt werden, wobei die begrenzte Dauer der Experimente hier keine Hinweise liefern konnte.

Eine problemlosere Eingabe und Wartung von Bahnen ist möglich, wenn die armseitigen Gelenkwinkel gemessen werden können. Alternativ muss untersucht werden, ob durch andere, leichter messbare Größen wie Beschleunigungen oder Motorströme die tatsächlichen Positionen der Armelemente beobachtbar sind und ob die Beobachter einigermaßen robust auslegbar sind.

Eine weitere Forderung an zukünftige Robotersteuerungen betrifft auch Roboter, bei denen aufgrund von anderer Bauart oder geringeren Beschleunigungen keine Nachgiebigkeiten merkbar sind. Das Problem ist die Integration der Vorsteuerung in die existierende Robotersteuerung, was zur Zeit nur über eine Sensorschnittselle möglich ist. Sowohl bei dieser Realisierung als auch bei zukünftig integrierten Versionen ist es erforderlich, dass die Bahnplanung prädiktiv erfolgt. Das bedeutet, dass die Interpolation (Feininterpolation) die

jeweils nächsten Punkte der Referenzbahn einige Schritte vor dem Erreichen ausrechnen muss. Ansonsten kann das vorgesteuerte Abfahren der Bahn nur um n_w Schritte verzögert erfolgen, was am Ende der Bahn einige Wartezyklen erfordert.

Bei Integration der Vorsteuerung in eine neue Robotersteuerung muss die Sensorschnittstelle neu definiert werden, da dann die Vorsteuerung nicht mehr in der Sensortask implementiert ist, wie bei der behelfsmäßigen Integration in bestehende Steuerungen. Während bisher die Sensorschnittstelle zwischen den beiden Funktionsblöcken des *idealen Roboters* aus Abb. 4.6 lag, gehört sie eigentlich an den Eingang. Dementsprechend reicht es nicht, nur die sensorgestützt modifizierten aktuellen Soll-Positionen $\mathbf{q}_d(k)$ zu verarbeiten. Stattdessen müssen die Vektoren $\mathbf{q}_d$ und τ_d^c über die nächsten n_w Abtastschritte erfasst werden.

Bei Realisierung der Sensortask nach Abb. 4.13 muss die Robotersteuerung die Referenztrajektorie $\mathbf{x}_r$ mit den erwarteten Sensorwerten $\mathbf{s}_r$ und der aktuellen Position $\mathbf{x}_a(k)$ liefern. Als Ausgang der Sensortask, also als Eingang des idealen Roboters kann man dann wahlweise die Achswerte $\mathbf{q}_d$ und τ_d^c oder die kartesischen Werte $\mathbf{x}_d$ und $\mathbf{F}_d$ vorsehen. Ggf. kann auch die Glättung dem *idealen Roboter* zugerechnet werden, sodass direkt $\mathbf{x}_s$ und $\mathbf{F}_s$ als Ausgang der Sensortask dienen.

Schließlich erfordert die neue Sensorschnittstelle ein Umdenken bei der Planung von Sensorsystemen. So müssen grundsätzlich prädiktive Sensoren vorgesehen werden, auch wenn aus Aufwandsgründen meist nur einfache Sensoren mit entsprechender Extrapolation der Signale eingesetzt werden.

Eine eigene Benutzerschnittstelle für eine adaptive Vorsteuerung ist dagegen bei zukünftigen Realisierungen nicht mehr zwingend notwendig. Dementsprechend kann die Vorsteuerung als eingebettetes System innerhalb der Robotersteuerung betrachtet werden. Das Ein- und Ausschalten von Trainingsvorgängen oder Kompensationen kann dabei über die normale Programmierumgebung erfolgen, bei KUKA Robotern also über die KRL (KUKA robot language), ähnlich wie Sensoren über die RSI (Robot sensor interface) angesprochen werden können, die bei der neuesten Version der Robotersteuerung KRC (KUKA robot controller) Bestandteil der KRL ist [9].

Kapitel 7

Zusammenfassung und Ausblick

In dieser Arbeit wurde die Frage untersucht, warum Industrieroboter bei schnell abgefahrenen Bahnen die vorgegebenen Trajektorien nur sehr ungenau einhalten können. Üblicherweise werden zur Behebung dieser Abweichungen beim Einprogrammieren einer Bahn von einem Bediener Korrekturen vorgegeben (Teach-in), durch die die resultierende Trajektorie die erlaubten Toleranzen einhält.

Es wurde eine Methode entwickelt, mit der das Teach-in durch eine adaptive Vorsteuerung ersetzt wird. Ziel war es, die dynamischen Bahnabweichungen zu minimieren. Als besonders vorteilhaft erwies sich dabei, dass das entwickelte Verfahren im Gegensatz zum klassischen Teach-in kein wiederholtes Abfahren derselben Bahn voraussetzt sondern sogar bei nur einmal auszuführenden Bewegungen die gewünschten Eigenschaften zeigt.

Außerdem wurde eine hierarchische Architektur vorgestellt, die den Einsatz der Vorsteuerung bei sensorbasierten Bewegungen erlaubt. Dadurch werden auch statische Ungenauigkeiten des Roboters oder Unsicherheiten in der Aufgabenbeschreibung ausgeglichen.

7.1 Hierarchische Architektur für schnelle sensorbasierte Bewegungen

Die Architektur besteht aus drei Ebenen, von denen die untere durch die vom Roboterhersteller vorgesehene industrielle Steuerung (Kaskadenregelung) reali-

siert ist. Die mittlere Ebene bildet eine adaptive Vorsteuerung. Beide Ebenen gemeinsam stellen einen *idealen Roboter* dar, dessen Entwurf bzw. Adaption unabhängig von der Aufgabe erfolgt. Die obere Ebene enthält die Bahnplanung unter Einbeziehung der Sensordatenverarbeitung. Diese Ebene ist im Gegensatz zu bekannten Verfahren der Rückkopplung von Sensordaten nicht vom realen Robotersystem oder seiner industriellen Steuerung beeinflusst. Sie wird stattdessen unter Einbeziehung der Aufgabenstellung und der verfügbaren Sensoren konfiguriert.

Schnittstelle zwischen der oberen und der mittleren Ebene sind Soll-Positionen und ggf. die dabei erwarteten Kontaktkräfte. Diese Daten werden von der oberen Ebene sowohl für den aktuellen Zeitschritt als auch für weitere, zukünftige Abtastschritte bereitgestellt, da nach Ansicht des Autors die Kenntnis der jeweils zukünftig gewünschten Bewegungen bei üblichen Industrierobotern die einzige Möglichkeit zu einer merklichen Erhöhung der Bahngenauigkeit ist.

7.2 Adaptiv vorausplanende Steuerung

Die mittlere Ebene des hierarchischen Systems ist durch eine vorausplanende Steuerung realisiert, da nur dadurch die erwähnte Information über die zukünftige gewünschte Bewegung verarbeitet werden kann.

Grundlage der Verbesserung der Bahngenauigkeit ist eine adaptive Struktur, die im Gegensatz zu anderen Adaptionsverfahren in drei Schritten arbeitet: Zuerst wird aufgrund einer Beispieltrajektorie ein grobes Streckenmodell identifiziert. Dieses Modell kann dann benutzt werden, um die Bewegungsanweisungen der Beispieltrajektorie zu optimieren. In einem dritten Schritt wird die optimierte Stellfolge schließlich auf einen Regler bzw. eine Vorsteuerung abgebildet.

Die Adaption erfolgt iterativ, d. h. eine zunächst suboptimale Vorsteuerung wird angewendet, um genauere Informationen zu gewinnen, die zu einer weiteren Reduktion des Bahnfehlers führen. Der Einfachheit halber wurde die Adaption offline demonstriert. Dabei konvergieren die Reglerparameter innerhalb weniger Iterationen. Eine andauernde Adaption ist also nicht nötig. Durch einen Kalman-Filter-basierten Entwurf des dreistufigen Schätzverfahrens garantieren die implementierten Algorithmen auch numerische Stabilität.

Zur Darstellung von nichtlinearen Effekten wurden Neuronale Netze vorgeschla-

gen. Als Netzttyp wurden mehrschichtige Perzeptrons ausgewählt, da sie sich bei großem Eingangsbereich am genauesten trainieren lassen. Trotzdem reichen die bekannten Verfahren zum Training nicht aus. Deswegen wurde ein Algorithmus auf der Basis des erweiterten Kalman Filters entwickelt.

7.3 Ergebnis

Das Verfahren wurde an vorgegebenen und an sensorgestützt erzeugten Bahnen erfolgreich eingesetzt. Schon bei linearer Vorsteuerung lässt sich eine Reduktion des an den Achsen gemessenen Bahnfehlers bei schnellen Bewegungen auf ein oder zwei zehntel Millimeter erreichen. Diese Verbesserung wird durch einmaliges Training des Roboters erreicht, z. B. bei der Installation. Die Neuronalen Netze verringern dabei die Störungen durch Kopplungen bis zum Faktor 5. Durch Training einer bahnspezifischen Steuerung lässt sich der verbleibende Fehler noch einmal halbieren. Die während einer Trajektorie auftretenden maximalen Bahnabweichungen lassen sich durch spezielle Wahl des Gütekriteriums selektiv reduzieren.

Im Gegensatz zu den in der Literatur durch Lernen erzielten leichten Verbesserungen, insbesondere durch Rückführung von Positions- oder Sensorwerten, ließ sich eine Reduktion der Bahnfehler um größenordnungsmäßig 90 % erreichen. Darüber hinaus sind die Vorsteuerungen im Gegensatz zu Lernverfahren bahnunabhängig gültig, erfordern also ein im Vergleich zur späteren Nutzung sehr geringes Training.

Im Vergleich zu modellbasierten Reglerentwürfen hat die adaptiv vorausplanende Steuerung insbesondere folgende Vorteile:

- Sie erfordert kein genaues dynamisches Modell des Roboters.
- Sie ist in bestehende Robotersteuerungen integrierbar.
- Der erforderliche Aufwand ist äußerst gering.

Die Arbeit zeigte aber auch Einschränkungen, in denen das Verfahren keine zufrieden stellenden Resultate bringt. Das betrifft vor allem die Bahngenauigkeit bei schnellem Abfahren von sensorgestützt modifizierten Bahnen. Es wurde

aber auch begründet, dass in dem Fall aufgrund von Stellgrößenbegrenzungen keine bessere Regelung möglich ist, sofern die Sensorik nicht prädiktiv in jedem Abtastschritt auch die zukünftigen Soll-Positionen bestimmen kann. Eine weitere Einschränkung betrifft die Kompensation von Kopplungen. So ist die Adaption der achsübergreifenden Einflüsse zwar nicht so bahnspezifisch wie eine Bahnsteuerung, die Verallgemeinerungsfähigkeit der Neuronalen Netze reicht allerdings nur für ähnlich Bahnen aus, was die praktische Anwendbarkeit sehr einschränkt. Schließlich lassen sich Schwingungen der Armelemente nur insoweit kompensieren, als sie messtechnisch erfasst werden. Eine Verbesserung dieses Punktes ist aufgrund der gewählten, allgemein verfügbaren Sensorschnittstelle kaum möglich.

Trotzdem ermöglicht das Verfahren den Einsatz von Industrierobotern für Aufgaben, die bisher aus Genauigkeitsgründen nicht oder nur bei eingeschränkter Ausführungsgeschwindigkeit automatisiert werden konnten. So erlauben die erreichten Verbesserungen auch bei hohen Bahngeschwindigkeiten das millimetergenaue Auftragen von Kleber oder Schneiden von Blechen. Die demonstrierte Genauigkeit ist Grundlage für die direkte Anwendung offline programmierter Bahnen. Darüber hinaus kann mit der vorgestellten hierarchischen Struktur auch bei grob geplanten Bahnen, die sensorgestützt verfeinert werden, mit ähnlich geringen Abweichungen vom gewünschten Verhalten gerechnet werden, sofern die Sensorik eine Prädiktion vorsieht. Im Fall einer visuell erfassten Soll-Bahn konnte die erforderliche Bahngenauigkeit bei voller Geschwindigkeit eingehalten werden.

7.4 Ausblick

Die vorliegende Arbeit hat die Verbesserung der Bahngenauigkeit bei den im Institut für Robotik und Mechatronik des DLR vorhandenen Industrierobotern mit den dort verfügbaren Messgeräten untersucht. Bei schnellen Robotern machten sich Nachgiebigkeiten im Getriebe bemerkbar. Das führte dazu, dass die industrielle Steuerung die tatsächliche Armposition nicht genau erfassen konnte. Für den Fall wurde die Adaption aufgrund der Werte eines externes Messgerätes gezeigt. Die Übertragbarkeit auf andere Bahnen oder Roboter, für die keine Messanordnung zur Verfügung steht, ist möglich, wobei mit Genauigkeitseinbußen von größenordnungsmäßig 0.1 mm für den mittleren Fehler gerechnet werden muss. Hier lassen sich mit anderen Messgeräten oder Robo-

tertypen weitere Erfahrungen gewinnen, wobei online auswertbare Messgeräte vorteilhaft sind, da sie im Unterschied zu den aufgeführten Experimenten auch bei der sensorgestützten Bahnplanung eingesetzt werden können.

Außerdem sind in einigen Punkten alternative Realisierungen möglich. Dazu gehört z. B. die Implementierung von Neuronalen Netzen mit radialen Basisfunktionen anstelle der mehrschichtigen Perzeptrons. Eine andere Möglichkeit der Weiterentwicklung des Verfahrens besteht in der Darstellung des Zeitverhaltens der Vorsteuerung. Bei hohen, von der industriellen Steuerung vorgegebenen Abtastraten zukünftiger Roboter ist die Zahl der Parameter der linearen Elemente bzw. die Zahl der Eingänge der nichtlinearen Anteile so groß, dass andere Darstellungen günstiger werden. Z. B. kann dann eine Repräsentation der Vorsteuerung durch orthonormale Basisfunktionen geeignet sein.

Schließlich wurde die Bahnplanungsebene nur exemplarisch betrachtet. In diesem Bereich können weitere Untersuchungen erfolgen, um in Abhängigkeit von Beispielaufgaben mit festgelegter Sensorkonfiguration ggf. effektivere Methoden zu entwickeln. Dazu gehört sicherlich auch die Prüfung von adaptiven Ansätzen, die in Bezug auf die Bahnplanung den Rahmen der Arbeit gesprengt hätte.

In dieser Arbeit konnten Methoden entwickelt und im Laborversuch beurteilt werden. Erfahrungen von industriellen Langzeituntersuchungen bei typischen Aufgaben unter Verwendung der entsprechenden Werkzeuge stehen bisher noch aus und werden der nächste Schritt zur Praxiseinführung sein.

Auf längere Sicht erfordert auch die Strategie der Nutzung prädiktiv vorausschauender Sensoren noch weitere Überlegungen zur Entwicklung und Integration geeigneter Komponenten.

In Zukunft wird es auch möglich sein, Roboter unter Verwendung eines vollständigen dynamischen Modells genauer zu regeln als dies mit den heute noch geltenden Aufwandsbeschränkungen für den Echtzeitbetrieb der Fall ist. Dann wird die Qualität der Regelung insbesondere von der Verfügbarkeit geeigneter dynamischer Modelle (z. B. zur Beschreibung von Reibungseinflüssen) und der (wirtschaftlichen) Identifizierbarkeit ihrer Parameter abhängen. Aufgrund von immer existierenden Restunsicherheiten der Modelle werden adaptive Verfahren als Ergänzung zukünftiger Steuerungen eingesetzt werden, um die verbleibenden Bahnfehler zu reduzieren. Die strukturellen Unterschiede gegenüber heutigen Industrieroboterregelungen sowie die qualitativ bessere Regelgüte solch zukünftiger Steuerungen werden allerdings eine Anpassung der für

heutige Steuerungen entwickelten adaptiven Verfahren erfordern.

Literaturverzeichnis

[1] The ABB group. http://www.abb.com/, 2001.

[2] ADEPT Roboter Übersicht. http://www.adept.de/produkte/roboter/ index.shtml/, 2000.

[3] H. A. Akeel and S. W. Holland. Product and technology trends for industrial robots. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 696–700, San Francisco, CA, April 2000.

[4] T. Alban and H. Janocha. Dynamic calibration of industrial robots with inertial measurement systems. In *Proc. European Control Conference (CD-version)*, Karlsruhe, Germany, August / Sept. 1999.

[5] A. Albu-Schäffer, K. Jöhl, and I. Schäfer. Force reflecting joystick. http://www.robotic.dlr.de/MMI/joystick, 1998.

[6] A. Albu-Schäffer and G. Hirzinger. State feedback controller for flexible joint robots: A globally stable approach implemented on DLR's lightweight robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1087–1093, Takamatsu, Japan, Oct./Nov. 2000.

[7] J. S. Albus. A new approach to manipulator control: The cerebellar model articulation controller (CMAC). *Trans. of the ASME, Journal of Dynamic Systems, Measurement, and Control*, pages 221–227, Sept. 1975.

[8] J. S. Albus, H. G. McCain, and R. Lumia. NASA/NBS standard reference model for telerobot control system architecture (NASREM). NIST Technical Note 1235, National Institute of Standards and Technology, Gaithersburg, MD 20899, 1989.

[9] Amatec Robotics GmbH, Germering. *FTCtrl Manual, Benutzerhandbuch für das RSI-Kraftregelungspaket zur KUKA Robotersteuerung KR C1 und KR C2 ab Softwarestand V4.0*, 2001.

[10] C. H. An, C. G. Atkeson, and J. M. Hollerbach. *Model-Based Control of a Robot Manipulator*. The MIT Press, Cambridge, Massachusetts, London, England, 1988.

[11] N. Andreff, R. Horaud, and B. Espiau. Robot hand-eye calibration using structure-from-motion. *The International Journal of Robotics Research*, 20(3), pages 228-248, Mar. 2001.

[12] K.-I. Arai and R. Nakano. Annealed RNN learning of finite state automata. In *Int. Conf. on Artificial Neural Networks ICANN '96*, pages 519–524, Bochum, July 1996. Lecture Notes in Computer Science, Vol. 1112, Springer.

[13] S. Arimoto, S. Kawamura, F. Miyazaki, and S. Tamaki. Learning control theory for dynamical systems. In *24th Conf. on Decision and Control*, pages 1375–1380, Ft. Lauderdale, FL, Dec. 1985.

[14] Advanced Realtime Tracking GmbH (ART). http://www.ar-tracking.de, 2001.

[15] C. G. Atkeson, E. W. Aboaf, J. McIntyre, and J. Reinkensmeyer. Model-based robot learning. In *Preprints 4th Int. Symp. of Robotic Research*, pages 9–16, Santa Cruz, CA, Aug. 1987.

[16] M. Bachiller, A. Adán, V. Feliu, and C. Cerrada. Well structured robot positioning control strategy for position based visual servoing. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2541–2546, Seoul, Korea, May 2001.

[17] J. Baeten and J. De Schutter. Combined vision / force control at corners in planar robotic contour following. In *Proc. IEEE/ASME Int. Conference on Advanced Intelligent Mechatronics*, Como, Italy, July 2001.

[18] J. Baeten, W. Verdonck, H. Bruyninckx, and J. De Schutter. Combining force control and visual servoing for planar contour following. *Machine Inteligence and Robotic Control*, 2(2), pages 69-75, 2000.

[19] D. H. Ballard. *An Introduction to Natural Computation*. A Bradford Book. The MIT Press, Cambridge, MA, London, England, 1997.

[20] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Trans.*

on Systems, Man, and Cybernetics, SMC-13(5), pages 834-846, Sept./Oct. 1983.

[21] A. Bernardino and J. Santos-Victor. Binocular tracking: Integrating perception and control. *IEEE Trans. on Robotics and Automation*, 15(6), pages 1080-1094, Dec. 1999.

[22] R. Bernhardt. MeDIaTe. http://www-robotics.ipk.fhg.de/rc/Mediate/mediate-sum.html, 1998. Fraunhofer Institut für Prodkutionsanlagen und Konstruktionstechnik.

[23] R. Bernhardt. RoboCal. http://www-robotics.ipk.fhg.de/rc/RoboCal/RoCal-sum-e.html, 1998. Fraunhofer Institut für Prodkutionsanlagen und Konstruktionstechnik.

[24] J. Bickendorf. Neue Anwendungsgebiete für Industrieroboter durch CAD-basierte Offline-Programmierung. In *Fachtagung Robotik 2000*, VDI Bericht Nr. 1552, Seiten 121-126, Berlin, Juni 2000.

[25] S. Bös. Learning curves of on-line and off-line training. In *Int. Conf. on Artificial Neural Networks ICANN '96*, pages 89–94, Bochum, July 1996. Lecture Notes in Computer Science, Vol. 1112, Springer.

[26] B. Brogliato, D. Rey, A. Pastore, and J. Barnier. Experimental comparison of nonlinear controllers for flexible joint manipulators. *The International Journal of Robotics Research*, 17(3), pages 260-281, Mar. 1998.

[27] B. Brunner, K. Landzettel, G. Schreiber, B.-M. Steinmetz, and G. Hirzinger. A universal task-level ground control and programming system for space robot applications - the MARCO concept and its applications to the ETS-VII project. In *Proc. 5th Int. Symp. on Artificial Intelligence, Robotics and Automation in Space*, ESTEC, Noordwijk, The Netherlands, June 1999.

[28] B. Brunner, K. Landzettel, B.-M. Steinmetz, and G. Hirzinger. - Telesensor-programming - A task-directed programming approach for sensor-based space robots. In *Proc. Int. Conference on Advanced Robotics*, pages 309–316, Sant Feliu de Guixols, Spain, Sept. 1995.

[29] B. Brunner, J. Vogel und G. Hirzinger. Aufgabenorientierte Fernprogrammierung von Robotern. *Automatisierungstechnik at*, 49(7), Seiten 312-319, 2001.

[30] H. Bruyninckx and O. Khatib. Gauss' principle and the dynamics of redundant and constrained manipulators. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2563–2568, San Francisco, CA, April 2000.

[31] E. Burdet and J. Luthiger. Adaptation of the visuo-motor coordination. In *IEEE Int. Conf. on Robotics and Automation*, pages 2656–2661, Minneapolis, Minnesota, April 1996.

[32] E. Burdet, B. Sprenger, and A. Codourey. Experiments in nonlinear adaptive control. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 537–542, Albuquerque, New Mexico, April 1997.

[33] B. Carlisle. Robot mechanism. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 701–708, San Francisco, CA, April 2000.

[34] Welcome to the CATIA users' page, by Dassault Systemes. http://www.catia.com/, 2001.

[35] E. Cervera and P. Martinet. Combining pixel and depth information in image-based visual servoing. In *Proc. Int. Conference on Advanced Robotics*, pages 445–450, Tokyo, Japan, Oct. 1999.

[36] E. Cervera and P. Martinet. Visual servoing with indirect image control and a predictable camera trajectory. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 381–386, Kyongju, Korea, Oct. 1999.

[37] F. Chaumette and E. Malis. 2 1/2 D visual servoing: a possible solution to improve image-based and position/based visual servoings. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 630–635, San Francisco, CA, April 2000.

[38] C.-C. Cheah and D. Wang. Learning impedance control for robotic manipulators. *IEEE Trans. on Robotics and Automation*, 14(3), pages 452-465, June 1998.

[39] C. L. P. Chen. A rapid supervised learning neural network for function interpolation and approximation. *IEEE Transactions on Neural Networks*, 7(5), pages 1220-1230, Sep. 1996.

[40] W. Chen, J. K. Mills, J. Chu, and D. Sun. A fuzzy compensator for uncertainty of industrial robots. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2968–2973, Seoul, Korea, May 2001.

[41] W.-H. Chen. Nonlinear PID predictive control of robotic manipulators. In *Preprints IFAC Symp. on Robot Control*, volume 1, Vienna, Austria, pages 217-222, Sept. 2000.

[42] Y. Chen. Webpages for iterative learning control (ilc) research. http://ilc.ee.nus.edu.sg/, 1999.

[43] Y. Chen. Updates on iterative learning control 2000, 2000. http://www.crosswinds.net/~ygchen/ilc2000/.

[44] Y. Chen, C. Wen, and M. Sun. A robust high-order P-type iterative learning controller using current iteration tracking error. *International Journal of Control*, 68(2), pages 331-342, 1997.

[45] J. Cheong, W. K. Chung, and Y. Youm. Fast suppression of vibration for multi-link flexible robots using parameter adaptive control. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 913–918, Maui, Hawaii, Oct./Nov. 2001.

[46] S. Chroust and M. Vincze. Comparison of predictive methods for vision-based control of motion. In *Preprints IFAC Symp. on Robot Control*, volume 2, Vienna, Austria, pages 693-698, Sept. 2000.

[47] S. Chroust, E. Zimmer, and M. Vincze. Pro and cons of control methods of visual servoing. In *Proc. 10th Int. Workshop on Robotics in Alpe-Adria-Danube Region (CD-version)*, Vienna, Austria, May 2001.

[48] M. K. Ciliz and K. S. Narendra. Adaptive control of robotic manipulators using multiple models and switching. *The International Journal of Robotics Research*, 15(6), pages 592-610, Dec. 1996.

[49] D. W. Clarke, C. Mohtadi, and P. S. Tuff. Generalized predictive control - part I. the basic algorithm. *Automatica*, 23(2), pages 137-148, 1987.

[50] R. Colbaugh, K. Glass, and E. Barany. Adaptive regulation of manipulators using only position measurements. *The International Journal of Robotics Research*, 16(5), pages 703-713, Oct. 1997.

[51] F. Conticelli and B. Allotta. Two-level visual control of dynamic look-and-move systems. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3784–3789, San Francisco, CA, April 2000.

[52] P. I. Corke and M. C. Good. Dynamic effects in visual closed-loop systems. *IEEE Trans. on Robotics and Automation*, 12(5), pages 671-683, Oct. 1996.

[53] P. I. Corke and S. A. Hutchinson. Real-time vision, tracking and control. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 622–629, San Francisco, CA, April 2000.

[54] R. Cortesão and R. Koeppe. Sensor fusion for skill transfer systems. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1014–1019, Kyongju, Korea, Oct. 1999.

[55] J. J. Craig. Adaptive control of manipulators through repeated trials. In *American Control Conference*, volume 3, San Diego, pages 1566-1573, June 1984.

[56] J. J. Craig. *Adaptive Control of Mechanical Manipulators.* Addison-Wesley Publishing Company, Reading, Massachusetts, 1988.

[57] J. J. Craig, P. Hsu, and S. S. Sastry. Adaptive control of mechanical manipulators. *The International Journal of Robotics Research*, 6(2), pages 16-28, 1987.

[58] L. Cromme. Lernverfahren in Feedforward Netzen. In *Beiträge zum Cottbuser Workshop 'Aspekte Neuronalen Lernens' CoWAN'98*, TU Cottbus, Okt. 1998.

[59] A. Datta and M.-T. Ho. A modified model reference adaptive control scheme for rigid robots. *IEEE Transactions on Robotics and Automation*, 12(3), pages 466-470, June 1996.

[60] V. R. de Angulo and C. Torras. Automatic recalibration of a space robot: An industrial prototype. In *Int. Conf. on Artificial Neural Networks ICANN '96*, pages 635–640, Bochum, July 1996. Lecture Notes in Computer Science, Vol. 1112, Springer.

[61] A. De Luca. Feedforward / feedback laws for the control of flexible robots. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 233–240, San Francisco, CA, April 2000.

[62] A. De Luca and P. Lucibello. A general algorithm for dynamic feedback linearization of robots with elastic joints. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 504–510, Leuven, Belgium, May 1998.

[63] A. De Luca, G. Paesano, and G. Ulivi. A frequency-domain approach to learning control: Implementation for a robot manipulator. In *4th IEEE Int. Symp. on Intelligent Control*, pages 66–71, Sep. 1989.

[64] A. De Luca, S. Panzieri, and G. Ulivi. Stable inversion control for flexible link manipulators. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 799–805, Leuven, Belgium, May 1998.

[65] J. De Schutter, D. Torfs, H. Bruyninckx, and S. Dutré. Invaraiant hybrid force / position control of a velocity controlled robot with compliant end effector using modal decoupling. *The International Journal of Robotics Research*, 16(3), pages 340-356, June 1997.

[66] J. Dehaes, P. Klewais, M. Nuttin, and P. Vanherck. Compensation of thermal deformations in machine tools with neural nets. In *Prepr. 2nd Workshop on Learning in Intelligent Manufacturing Systems*, pages 746–756, 1995.

[67] Welcome to Delmia. http://www.delmia.com/, 2001.

[68] S. Demey and J. De Schutter. Identification of second order surface geometry with a force controlled robot. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 473–480, Osaka, Japan, Nov. 1996.

[69] E. D. Dickmanns. An expectation-based, multi-focal, saccadic (ems) (inertially stabilized dynamic) vision system for vehicle guidance. In *Int. Symp. of Robotics Research (ISRR'99)*, pages 334–339, Snowbird, Utah, Oct. 1999.

[70] DIN EN ISO 9283: Industrieroboter - Leistungskenngrößen und zugehörige Prüfmethoden (ISO 9283: 1998). Beuth Verlag, Berlin, Wien, Zürich, Ausgabe 1999-05.

[71] DIN EN ISO 9787: Industrieroboter - Koordinatensysteme und Bewegungsnomenklaturen (ISO 9787: 1999). Beuth Verlag, Berlin, Wien, Zürich, Ausgabe 2000-07.

[72] DIN EN ISO 9946: Industrieroboter - Darstellung charakteristischer Eigenschaften (ISO 9946: 1999). Beuth Verlag, Berlin, Wien, Zürich, Ausgabe 2000-07.

[73] M. Dresselhaus and J. Held. Entwicklungskonzept, Realisierung und Erprobung modellbasierter Regelungsverfahren für innovative Industrieroboter-systeme. In *Fachtagung Robotik 2000*, VDI Bericht Nr. 1552, Seiten 179-184, Berlin, Juni 2000.

[74] P. Drouet, S. Dubowsky, and C. Mavroidis. Compensation of geometric and elastic deflection errors in large manipulators based on experimental measurements: Application to a high accuracy medical manipulator. In J. Lenarčič and M. L. Husty, editors, *Advances in Robot Kinematics: Analysis and Control*, pages 513–522. Kluwer Academic Publishers, 1998.

[75] Dynalog, Inc. *3D CompuGauge: User's Manual*, 1997. Version 2.7.

[76] A. Elfving and U. Kirchhoff. Design methodology for space automation and robotic systems. *ESA Journal*, 15, pages 149-164, 1991.

[77] E. Ersü. On the application of associative neural network models to technical control problems. In *Localization and Orientation in Biology and Engineering*, pages 90–93, Tübingen, 1984. 8. Kybernetik Kongreß, März 1983, Springer.

[78] E. Ersü and H. Tolle. A new concept for learning control inspired by brain theory. In *9th IFAC World Congress*, volume 2, Budapest, Hungary, pages 1039-1044, July 1984.

[79] N. Faiz and S. K. Agrawal. Trajectory planning of robots with dynamics and inequalities. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3977–3983, San Francisco, CA, April 2000.

[80] FANUC Robotics. http://www.fanucrobotics.com/, 2001.

[81] M. Ferch, J. Zhang, and A. Knoll. Robot skill transfer based on b-spline fuzzy controllers for force control tasks. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1164–1169, Detroit, Michigan, May 1999.

[82] G. Ferretti, G. Magnani, P. Rocco, F. Cecconello, and G. Rossetti. Impedance control for industrial robots. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 4028–4033, San Francisco, CA, April 2000.

[83] W. Finnoff, F. Hergert, and H. G. Zimmermann. Extended regularization methods for nonconvergent model selection. In S. Hanson, J. Cowan, and

C. Giles, editors, *Advances in Neural Information Processing Systems*, volume 5. San Mateo, pages 228-235, 1993.

[84] O. Föllinger. *Regelungstechnik.* Hüthig Verlag, Heidelberg, 8. Auflage, 1994.

[85] G. E. Forsythe und C. B. Moler. *Computerverfahren für lineare algebraische Systeme.* Oldenbourg, 1979.

[86] P. M. Frank. *Empfindlichkeitsanalyse dynamischer Systeme.* Oldenbourg, 1976.

[87] C. W. Frey, M. Sajidman und H.-B. Kuntze. Ein neuro-adaptives Regelungskonzept mit on-line-fähigem Kalman-Filter Lerverfahren für stochastisch gestörte nichtlineare Prozesse. In *Computational Intelligence : Neuronale Netze, Evolutionäre Algorithmen, Fuzzy Control im industriellen Einsatz*, VDI Bericht Nr. 1381, Seiten 381-392, Berlin, März 1998.

[88] J. A. Gangloff, M. de Mathelin, and G. Abba. 6 dof high speed dynamic visual servoing using GPC controllers. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2008–2013, Leuven, Belgium, May 1998.

[89] J. A. Gangloff and M. F. de Mathelin. High speed visual servoing of a 6 DOF manipulator using MIMO predictive control. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3751–3756, San Francisco, CA, April 2000.

[90] C. Gégout. Improvement of multilayer perceptron training with an evolutionary initialization. In *Int. Conf. on Artificial Neural Networks ICANN '95*, volume 2, Paris, France, pages 153-158, Oct. 1995.

[91] B. Geyer und H. Rake. Prädiktive Regelung von Roboterantrieben mit linearen Zustandsraummodellen. In *GMA-Kongress „Mess- und Automatisierungstechnik“*, VDI Bericht Nr. 1282, Seiten 367-376, Baden-Baden, Sept. 1996.

[92] G. Goos. *Vorlesungen über Informatik 4 - Paralleles Rechnen und nichtanalytische Lösungsverfahren.* Springer, 1998.

[93] G. Grunwald. *Aufgabenorientierte Planung kooperierender Sensoren.* Dissertation, Universität Karlsruhe, 1994.

[94] S. Gunnarsson and M. Norrlöf. Iterative learning control of a flexible mechanical system using accelerometers. In *Preprints IFAC Symp. on Robot Control*, volume 2, Vienna, Austria, pages 573-578, Sept. 2000.

[95] Q. P. Ha, A. Bonchis, D. C. Rye, and H. F. Durrant-Whyte. Variable structure systems approach to friction estimation and compensation. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3543–3548, San Francisco, CA, April 2000.

[96] S.-H. Han, W. H. Seo, K. S. Yoon, and M.-H. Lee. Real-time control of an industrial robot using image-based visual servoing. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1762–1767, Kyongju, Korea, Oct. 1999.

[97] B. Heinrichs and N. Sepheri. A limitation of position based impedance control in static force regulation: Theory and experiments. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2165–2170, Detroit, Michigan, May 1999.

[98] T. Heskes and W. Wiegerinck. Presentation order and on-line learning. In *Int. Conf. on Artificial Neural Networks ICANN '95*, volume 1, Paris, France, pages 223-228, Oct. 1995.

[99] J. Hespanha, Z. Dodds, G. D. Hager, and A. S. Morse. What can be done with an uncalibrated stereo system? In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1366–1372, Leuven, Belgium, May 1998.

[100] J. Hesselbach, M. Frindt, H. Kerle, and I. Pietsch. PORTYS - a machine concept with parallel structure for precise pick & place operations at high speed. In *Proc. 10th Int. Workshop on Robotics in Alpe-Adria-Danube Region (CD-version)*, Vienna, Austria, May 2001.

[101] G. Hirzinger. The magellan spacemouse - its roots are in space. http://www.robotic.dlr.de/MMI/sm.html, 1998.

[102] G. Hirzinger. Adaptiv sensorgeführte Roboter mit besonderer Berücksichtigung der Kraft-Momenten-Rückkopplung. *Robotersysteme*, 1, Seiten 161-171, 1985.

[103] G. Hirzinger, J. Butterfaß, M. Fischer, M. Grebenstein, M. Hähnle, H. Liu, I. Schaefer, and N. Sporer. Mechatronics approach to the design of lightweight arms and multifingered hands. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 46–54, San Francisco, CA, April 2000.

[104] N. Hogan. Impedance control: An approach to manipulation, parts I-III. *Trans. of the ASME, Journal on Dynamics, Systems, Measurement, and Control*, 107(1), pages 1-24, 1985.

[105] M. Honegger, R. Brega, and G. Schweitzer. Application of a nonlinear adaptive controller to a 6 dof parallel manipulator. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1930–1935, San Francisco, CA, April 2000.

[106] H. Hoyer, G.-H. Wen und T. Notheis. Prädiktive Regelung und Modellbildung von Robotern mit dem GPIA-Identifikationsverfahren. In *Fachtagung Intelligente Steuerung und Regelung von Robotern*, VDI Bericht Nr. 1094, Seiten 275-284, Langen, Nov. 1993.

[107] S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5), pages 651-670, Oct. 1996.

[108] R. Isermann. *Digitale Regelsysteme Band II.* Springer, 1987.

[109] R. Isermann. *Identifikation dynamischer Systeme Band I und II.* Springer, 2. Auflage, 1992.

[110] A. Jacubasch, H.-B. Kuntze, C. Arber und J. Richalet. Anwendung eines neuen Verfahrens zur schnellen und robusten Positionsregelung von Industrierobotern. *Robotersysteme*, 3, Seiten 129-138, 1987.

[111] S. Jagannathan and F. L. Lewis. Multilayer discrete-time neural-net controller with guaranteed performance. *IEEE Transactions on Neural Networks*, 7(1), pages 107-130, 1996.

[112] R. Jamisola, M. Ang, Jr., T. M. Lim, O. Khatib, and S. Y. Lim. Dynamics identification and control of an industrial robot. In *Proc. Int. Conference on Advanced Robotics*, pages 323–328, Tokyo, Japan, Oct. 1999.

[113] M. Jankovic. Observer based control for elastic joint robots. *IEEE Trans. on Robotics and Automation*, 11(4), pages 618-623, Aug. 1995.

[114] M. I. Jordan and D. E. Rumelhart. Forward models: Supervised learning with a distal teacher. *Cognitive Science*, 16, pages 307-354, 1992.

[115] S. Jung, S. B. Yim, and T. C. Hsia. Experimental studies of neural network impedance force control for robot manipulators. In *Proc. IEEE*

Int. Conference on Robotics and Automation, pages 3453–3458, Seoul, Korea, May 2001.

[116] L. P. Kaelbling. On reinforcement learning for robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1319–1320, Osaka, Japan, Nov. 1996.

[117] P. Kaever. *Selbsttätige Verbesserung der Regelung eines sensorgeführten Industrieroboters.* Diplomarbeit, Universität Karlsruhe, Institut für Regelungs- und Steuerungssysteme, 1987.

[118] M. Kaiser. Time-delay neural networks for control. In *Preprints 4th IFAC Symp. on Robot Control*, pages 967–972, Capri, Italy, Sept. 1994.

[119] R. E. Kalman. A new approach to linear filtering and prediction problems. *Trans. of the ASME, Journal of Basic Eng.*, 82, pages 34-45, 1960.

[120] D. Katić and S. Stanković. Fast learning algorithms for training of feedforward multilayer perceptrons based on extended Kalman filter. In *IEEE Int. Conf. on Neural Networks ICNN'96*, pages 196–201, Washington, June 1996.

[121] D. Katić and M. Vukobratović. Learning control algorithms for robot contact task using feedforward neural networks. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, volume 3, Pittsburgh, Pennsylvania, pages 522-527, Aug. 1995.

[122] T. Kavli. Nonuniformly partitioned piecewise linear representation of continuous learned mappings. In *IEEE Int. Workshop on Intelligent Motion Control*, pages 115–122, Bogazici University, Istanbul, Aug. 1990.

[123] H. Kawasaki, T. Bito, and K. Kanzaki. An efficient algorithm for the model-based adaptive control of robotic manipulators. *IEEE Transactions on Robotics and Automation*, 12(3), pages 496-501, June 1996.

[124] M. Kawato. Computational schemes and neural network models for formation and control of multijoint arm trajectory. In W. T. Miller III, R. S. Sutton, and P. J. Werbos, editors, *Neural Networks for Control*, pages 197–228, Cambridge, Massachusetts, London, England, 1990. The MIT Press.

[125] W. Khalil, G. Garcia, and J.-F. Delagarde. Calibration of the geometric parameters of robots without external sensors. In *IEEE Int. Conf. on Robotics and Automation*, pages 3039–3044, Nagoya, Japan, May 1995.

[126] W. Khalil and M. Gautier. Modeling of mechanical systems with lumped elasticity. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3965–3970, San Francisco, CA, April 2000.

[127] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation*, RA-3, pages 43-53, 1987.

[128] H. Kiendl. *Fuzzy Control methodenorientiert.* Oldenbourg, München, 1997.

[129] Y. H. Kim and F. L. Lewis. Output feedback control of rigid robots using dynamic neural networks. In *IEEE Int. Conf. on Robotics and Automation*, pages 1923–1928, Minneapolis, Minnesota, April 1996.

[130] FZ Karlsruhe - The KISMET simulation homepage. http://www-kismet.iai.fzk.de/KISMET/kismet.html, 2000.

[131] R. Koeppe, A. Breidenbach, and G. Hirzinger. Skill representation and acquisition of compliant motions using a teach device. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, Osaka, Japan, Nov 1996.

[132] R. Kofahl. *Robuste Parameteradaptive Regelungen.* Fachberichte Messen - Steuern - Regeln, Band 19. Springer, 1988.

[133] T. Kolb. Lernen in Rekurrenten Neuronalen Netzen? In *Beiträge zum Cottbuser Workshop 'Aspekte Neuronalen Lernens' CoWAN'98*, TU Cottbus, Okt. 1998.

[134] S. K. Kopparapu and P. Corke. The effect of measurement noise on intrinsic camera calibration parameters. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1281–1286, Detroit, Michigan, May 1999.

[135] D. Kragić and H. I. Christensen. Cue integration for visual servoing. *IEEE Trans. on Robotics and Automation*, 17(1), pages 18-27,Feb. 2001.

[136] Robot solutions. Krypton Electronic Engineering, Leuven, Belgium, 1998.

[137] KUKA Roboter GmbH, Augsburg. *Robotersteuerung KRC1 und Roboter KR6/1*, 1996. Interne Information.

[138] KUKA Schweißanlagen und Roboter GmbH, Augsburg. *Betriebsanleitung IR 363/6.0 und Rechnerteil KRC32*, Stand 1995.

[139] Willkommen bei KUKA Roboter GmbH. http://www.kuka-roboter.de/, 2001.

[140] K. K. Kumbla and M. Jamshidi. Neural network based identification of robot dynamics used for neuro-fuzzy controller. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1118–1123, Albuquerque, New Mexico, April 1997.

[141] H.-B. Kuntze, M. Sajidman, and A. Jacubasch. A fuzzy-logic concept for highly fast and accurate position control of industrial robots. In *IEEE Int. Conf. on Robotics and Automation*, pages 1184–1190, Nagoya, Japan, May 1995.

[142] S. Kwon and W. K. Chung. An improved pertubation attenuation method for motion control of robotic systems. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2060–2065, Maui, Hawaii, Oct./Nov. 2001.

[143] T. Lahdhiri and H. A. ElMaraghy. Optimal nonlinear position tracking control of a two-link flexible-joint robot manipulator. In *Preprints Int. Symp. on Experimental Robotics,ISER'97*, pages 407–418, Barcelona, Spain, June 1997.

[144] F. Lange. Der Assoziativspeicher ME7. Forschungsbericht DFVLR-FB 86-31, DFVLR, Institut für Dynamik der Flugsysteme, Oberpfaffenhofen, Juni 1986.

[145] F. Lange. Der Assoziativspeicher ME8. Institutsbericht IB 515-87-13, DFVLR, Institut für Dynamik der Flugsysteme, Oberpfaffenhofen, Mai 1987.

[146] F. Lange. Lösung von gestörten linearen Gleichungssystemen, bei denen die Koeffizientenmatrix eine bandähnliche Struktur hat, mit Anwendung auf die Steuerung eines dynamischen Systems. Institutsbericht IB 515-87-15, DFVLR, Institut für Dynamik der Flugsysteme, Oberpfaffenhofen, Aug. 1987.

[147] F. Lange. A learning concept for improving robot force control. In *Proc. IFAC Symposium on Robot Control*, pages 79.1–79.6, Karlsruhe, Germany, Oct. 1988.

[148] F. Lange. Schätzung und Darstellung von mehrdimensionalen Abbildungen. DLR Mitteilung DLR-Mitt. 90-06, DLR, Institut für Dynamik der Flugsysteme, Oberpfaffenhofen, Jan. 1990.

[149] F. Lange. Schätzung von Parametern linearer und nichtlinearer Gleichungen. Institutsbericht IB 515-90-7, DLR, Institut für Dynamik der Flugsysteme, Oberpfaffenhofen, Nov. 1990.

[150] F. Lange. Vorverarbeitung von Kraft- / Momentenmeßwerten. Institutsbericht IB 515-92-18, DLR, Institut für Robotik und Systemdynamik, Oberpfaffenhofen, Juli 1992.

[151] F. Lange. Fast and accurate training of multilayer perceptrons using an extended Kalman filter (EKFNet), Sept. 1995. http://www.robotic.dlr.de/Friedrich.Lange/.

[152] F. Lange and G. Hirzinger. Iterative self-improvement of force feedback control in contour tracking. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1399–1404, Nice, France, April 1992.

[153] F. Lange and G. Hirzinger. Adaptive minimization of the maximal path deviations of industrial robots. In *Proc. European Control Conference (CD-version)*, Karlsruhe, Germany, August / Sept. 1999.

[154] F. Lange and G. Hirzinger. Learning accurate path control of industrial robots with joint elasticity. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2084–2089, Detroit, Michigan, May 1999.

[155] F. Lange and G. Hirzinger. A universal sensor control architecture considering robot dynamics. In *Proc. Int. Conf. on Multisensor Fusion and Integration for Intelligent Systems*, pages 277–282, Baden-Baden, Germany, August 2001.

[156] F. Lange, J. Langwald, and G. Hirzinger. Predictive feedforward control for high speed tracking tasks. In *Proc. European Control Conference (CD-version)*, Karlsruhe, Germany, August / Sept. 1999.

[157] J. H. Lee, B. H. Lee, S. M. Lee, and C. Y. Chung. Preshaped trajectory command for fast repetative PTP motion of PD-controlled flexible joint

manipulators. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1546–1552, Grenoble, France, Sept. 1997.

[158] S. Lee and H. Asada. A pertubation / correlation method for force guided robot assembly. *IEEE Trans. on Robotics and Automation*, 15(4), pages 764-773, Aug. 1999.

[159] Leica Geosystems Lösungen und Systeme für Positionsbestimmung, Vermesung, Kartierung, Navigation, Industrievermesung, Verteidigung und Sicherheit, Maschinensteuerung, Kataster, Bauwesen, Photogrammetrie, Bergbau und andere Vermessungs- und Messanwendungen. http://www.leica-geosystems.com/, 2001.

[160] M. Leš und R. Svečko. Verbesserung der nichtparametrischen prädiktiven Regelung von nichtminimalphasigen Regelstrecken. *Automatisierungstechnik at*, 49(3), Seiten 132-137, 2001.

[161] W. Licheng, S. Zengqi, and S. Fuchum. Neural networks control structure for manipulators with flexible last link. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 2404–2408, Maui, Hawaii, Oct./Nov. 2001.

[162] P. Lucibello. A learning algorithm for improved hybrid force control of robot arms. *IEEE Trans. on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 28(2), pages 241-244, Mar. 1998.

[163] Z.-W. Luo, M. Ito, A. Kato, and K. Ito. Nonlinear robust control for robot compliant manipulation on dynamic environments. *Advanced Robotics*, 10(2), pages 213-227, 1996.

[164] R. Maaß, V. Zahn, M. Dapper, and R. Eckmiller. Hard contact surface tracking for industrial manipulators with (SR) position based force control. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1481–1486, Detroit, Michigan, May 1999.

[165] C. J. B. Macnab and G. M. T. D'Eleuterio. Stable, on-line learning using CMACs for neuroadaptive tracking control of flexible-joint manipulators. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 511–517, Leuven, Belgium, May 1998.

[166] P. Martinet and J. Gallice. Position based visual servoing using a nonlinear approach. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 531–536, Kyongju, Korea, Oct. 1999.

[167] M. T. Mason. Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11, pages 418-432, 1981.

[168] T. H. Massie and J. K. Salisbury. The PHANToM haptic interface: A device for probing virtual objects. In *ASME Winter Annual Meeting, Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 1994.

[169] P. S. Maybeck. *Stochastic Models, Estimation and Control, Volume 1.* Mathematics in Science and Engineering, Volume 141. Academic Press, New York, San Francisco, London, 1979.

[170] M. A. Meggiolaro, P. C. L. Jaffe, and S. Dubowsky. Achieving fine absolute positioning accuracy in large powerful manipulators. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2819–2824, Detroit, Michigan, May 1999.

[171] Y. Mezouar and F. Chaumette. Path planning in image space for robust visual servoing. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2759–2764, San Francisco, CA, April 2000.

[172] S. Miesbach. *Bahnführung von Robotern mit Neuronalen Netzen.* Dissertation, Technische Universität München, 1995.

[173] L. Mihaylova, H. Bruyninckx, J. De Schutter, and E. Staffetti. Planar contour tracking in the presence of pose and model errors by Kalman filterung techniques. In *Proc. IEEE Int. Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 329–334, Baden-Baden, August 2001.

[174] L. Mihaylova, T. Lefebvre, E. Staffetti, H. Bruyninckx, and J. De Schutter. Tracking contact transitions during force-controlled compliant motion using an interacting multiple model estimator. In *Proc. IEEE Int. Conference on Advanced Robotics*, pages 665–670, Budapest, Hungary, August 2001.

[175] J. Militzer und H. Tolle. Vertiefungen zu einem Teilbereiche der menschlichen Intelligenz imitierenden Regelungsansatz. In *Jahrbuch der Deutschen Gesellschaft für Luft- und Raumfahrt DGLR*, Band I, Seiten 18-27, 1986.

[176] W. T. Miller III, F. H. Glanz, and L. G. Kraft III. Application of a general learning algorithm to the control of robotic manipulators. *The International Journal of Robotics Research*, 6(2), pages 84-98, 1987.

[177] H. Miyamoto, M. Kawato, T. Setoyama, and R. Suzuki. Feedback-error-learning neural network for trajectory control of a robotic manipulator. *Neural Networks*, 1, pages 251-265, 1988.

[178] T. Miyazaki, K. Maekawa, and T. Bamba. Compensation of positioning errors of industrial robot using neural network. In *23rd Symp. on Industrial Robots*, pages 377–382, Barcelona, Spain, Oct. 1992.

[179] J.-H. Moon, T.-Y. Doh, and M. J. Chung. An iterative learning control scheme for manipulators. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 759–765, Grenoble, France, Sept. 1997.

[180] K. L. Moore. Iterative learning control: An expository overview. In B. N. Datta, editor, *Applied and Computational Control, Signals, and Circuits*, pages 151–214, Boston, 1999. Birkhäuser.

[181] K. L. Moore and J.-X. Xu. Special issue on iterative learning control. *Int. Journal of Control*, 73(10), pages 819-823, 2000. (other papers on ILC see pp. 824-999).

[182] G. Morel, E. Malis, and S. Boudet. Impedance based combination of visual and force control. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1743–1748, Leuven, Belgium, May 1998.

[183] D. M. Morris, R. Hebbar, and W. S. Newman. Force guided assemblies using a novel parallel manipulator. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 325–330, Seoul, Korea, May 2001.

[184] P. C. Müller. Robuste Positionsregelung von Robotern mittels Nichtlinearitäten-Schätzung und -Kompensation. *Automatisierungstechnik at*, 48(6), pages 289-295, 2000.

[185] C. Natale, B. Siciliano, and L. Villani. Spatial impedance control of redundant manipulators. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1788–1793, Detroit, Michigan, May 1999.

[186] O. Nelles. LOLIMOT - Lokale, lineare Modelle zur Identifikation nichtlinearer, dynamischer Systeme. *Automatisierungstechnik at*, 45(4), Seiten 163-174, 1997.

[187] O. Nelles, S. Ernst, M. Ayoubi, A. Schumann und K.-H. Lachmann. Identifikation mit einem Hammersteinmodell und neuronalen Netzen am Beispiel eines Abgasturboladers. *Automatisierungstechnik at*, 46(9), Seiten 411-419, 1998.

[188] O. Nelles, S. Ernst und R. Isermann. Neuronale Netze zur Identifikation nichtlinearer, dynamischer Systeme: Ein Überblick. *Automatisierungstechnik at*, 45(6), Seiten 251-262, 1997.

[189] B. Nemec and L. Žlajpah. Implementation of force control on redundant robot. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1314–1319, Victoria, B. C., Canada, Oct. 1998.

[190] W. S. Newman, M. S. Branicky, H. A. Podgurski, S. Chhatpar, L. Huang, J. Swaminathan, and H. Zhang. Force-responsive robotic assembly of transmission components. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2096–2102, Detroit, Michigan, May 1999.

[191] M. Norrlöf and S. Gunnarson. Some results on iterative learning control with disturbances. Technical Report LiTH-ISY-R-2020, Linköping University, Sweden, 1997. available at http://www.control.isy.liu.se/.

[192] M. M. Olsen and H. G. Petersen. A new method for estimating parameters of a dynamic robot model. *IEEE Trans. on Robotics and Automation*, 17(1), pages 95-100, Feb. 2001.

[193] M. Oya, Y. Onizuka, and K. Sato. Robust tracking control of rigid link flexible joint robots using only joint position and velocity measurements. In *Proc. Int. Conference on Advanced Robotics*, pages 391–396, Tokyo, Japan, Oct. 1999.

[194] H. Ozaki, T. Shimogawa, and C.-J. Lin. A feedback control gain tuning for mechatronic systems by iterative trials. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1894–1899, Maui, Hawaii, Oct./Nov. 2001.

[195] M. Pandit und K. Buchheit. Iterativ lernende Regelung zyklischer Produktionsprozesse. *Automatisierungstechnik at*, 44(1), pages 21-31, 1996.

[196] J. A. Piepmeier, G. V. McMurray, and H. Lipkin. Tracking a moving target with model independent visual servoing: a predictive estimation approach. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2652–2657, Leuven, Belgium, May 1998.

[197] K. B. Pimenta, J. P. de Souza, J. M. Rosário, and D. Dumur. Control of robotic joints with generalized predictive control (GPC). In *Proc. 10th Int. Workshop on Robotics in Alpe-Adria-Danube Region (CD-version)*, Vienna, Austria, May 2001.

[198] T. Poggio and F. Girosi. Networks for approximation and learning. *Proc. of the IEEE*, 78, pages 1481-1497, 1990.

[199] P. Poignet and M. Gautier. Comparison of weighted least squares and extended kalman filtering methods for dynamic identification of robots. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3622–3627, San Francisco, CA, April 2000.

[200] Polhemus Inc. Colchester, Vermont, USA. *3SPACE ISOTRAK II User's Manual*, 1993.

[201] D. Popescu, D. Selisteanu, and C. Ionete. Non-model based neural robot control. In *Proc. 10th Int. Workshop on Robotics in Alpe-Adria-Danube Region (CD-version)*, Vienna, Austria, May 2001.

[202] G. Pritschow, A. Horn und K. Grefen. Dynamisches Verhalten und Grenzen sensorgeführter Industrieroboter mit vorausblickendem Sensor. *Robotersysteme*, 8, Seiten 155-161, 1992.

[203] G. Pritschow, H. Klingel, M. Bauder und A. Horn. Erhöhung der Bahngenauigkeit von Industrierobotern. *Robotersysteme*, 8, Seiten 162-170, 1992.

[204] M. Prüfer. *Reibungsanalyse und Identifikation von Dynamikparametern bei direktangetriebenen und getriebebehafteten Robotern.* Fortschritte der Robotik, Band 1. Shaker Verlag, 1996.

[205] M. Prüfer and C. Schmidt. Practical aspects of robot dynamics. In *3rd Int. Symp. on Methods and Models in Automation and Robotics*, pages 971–976, Sept. 1996.

[206] M. Prüfer, C. Schmidt, and F. Wahl. Identification of robot dynamics with differential and integral models: A comparison. In *IEEE Int. Conf. on Robotics and Automation*, pages 340–345, San Diego, California, May 1994.

[207] M. H. Raibert and J. J. Craig. Hybrid position / force control of manipulators. *Trans. of the ASME, Journal on Dynamics, Systems, Measurement, and Control*, 102, pages 126-133, 1981.

[208] Reis Robotics - die Welt der Roboter. http://www.reisrobotics.com/, 2001.

[209] J. Richalet, A. Rault, J. L. Testud, and J. Papon. Model predictive heuristic control: Applications to industrial processes. *Automatica*, 14, pages 413-428, 1978.

[210] C. Robl and G. Färber. System architecture for synchronizing, signal level fusing, simulating and implementing sensors. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1639–1644, San Francisco, CA, April 2000.

[211] F. Rosenblatt. *Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms.* Spartan Books, Washington, DC, 1962.

[212] D. Rumelhart, J. McCelland, and the PDP Research Group. *Parallel Distributed Processing , Vol. 1.* The MIT Press, Cambridge, Massachusetts, London, England, 1986.

[213] I. Schäfer. DLR Leichtbauroboter Homepage. http://www.robotic.dlr.de/LBR, 1999.

[214] C. Schittenkopf, G. Deco, and W. Brauer. A novel pruning method to avoid overfitting in feed forward networks. In *Int. Conf. on Artificial Neural Networks ICANN '95*, volume 2, Paris, France, Oct. pages 437-442, 1995.

[215] D. Schmid, H. Nowak und E. Michalak. Dynamische Eigenschaften programmgeführter und sensorgeführter Industrieroboter. Technical report, Kernforschunsgzentrum Karlsruhe KfK-PFT 127, 1986.

[216] G. Schneider. Neue Bedien- und Programmierverfahren für Robotersteuerungen. In *VDI/VDE-GMA-Fachausschuss „Steuerung und Regelung von Robotern“*, Zürich, Schweiz, Jan. 1997.

[217] G. Schram, F. X. van der Linden, B. J. A. Kröse, and F. C. A. Groen. Predictive robot control with neural networks. *Intelligent Autonomous Systems*, pages 117–122, 1995.

[218] K. Schröer, editor. *State, Obstacles and Requirements of Performance Criteria Application*, volume 1, Report and Conclusions of the IRIS Survey of *Improvement of Robot Industrial Standardisation (IRIS), Programme on Standard, Measurements and Testing, Project-No SMT4-CT95-*

2013. Fraunhofer-Institut für Produktionsanlagen und Konstruktionstechnik (IPK), Berlin, 1997.

[219] K. Schröer, editor. *Handbook on Robot Perfromance - Testing and Calibration*. Improvement of Robot Industrial Standardisation (IRIS), Programme on Standard, Measurements and Testing, Project-No SMT4-CT95-2013. Fraunhofer IRB Verlag, 1998.

[220] H. R. Schwarz, H. Rutishauser und E. Stiefel. *Matrizen-Numerik*. Leitfäden der angewandten Mathematik und Mechanik, Band 11. Teubner Verlag, Stuttgart, 1968.

[221] L. Sciavicco and B. Siciliano. *Modeling and Control of Robot Manipulators*. Electrical and Computer Engineering Series. McGraw-Hill, 1996.

[222] Sensable technologies - the magic of touch. http://www.sensable.com/, 2001.

[223] H. Seraji and R. Colbaugh. Force tracking in impedance control. *The International Journal of Robotics Research*, 16(1), pages 97-117, Feb. 1997.

[224] B. Siciliano and L. Villani. *Robot Force Control*. Kluwer Academic Publishers, 1999.

[225] Siemens AG. *Sirotec RCM2 und RCM3 Handbücher*, 1989.

[226] S. Singhal and L. Wu. Training feed-forward networks with the extended Kalman algorithm. In *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 1187–1190, Glasgow, Scotland, 1989.

[227] SNNS (Stuttgart neural network simulator) user manual, version 4.2. A. Zell and others, University of Stuttgart, Institute for Parallel and Distributed High Performance Systems (IPVR), University of Tübingen, Wilhelm-Schickard-Institute for Computer Science, 1998. ftp://ftp.informatik.uni-stuttgart.de/pub/SNNS/.

[228] B. Song and A. J. Koivo. Neural network model based control of a flexible link manipulator. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 812–817, Leuven, Belgium, May 1998.

[229] Staubli textile, connector and robotics. http://www.staubli.com/, 2001.

[230] J. Stelter. *Verbesserung des Positionierverhaltens und der Bahntreue eines Industrieroboters durch den Einsatz von Beschleunigungssensoren.* Dissertation, Universität Karlsruhe, 2001.

[231] J. Suchý, M. Baroš, and J. Škrinárová. Investigation of the CMAC neural network for robotics. In *Proc. 10th Int. Workshop on Robotics in Alpe-Adria-Danube Region (CD-version)*, Vienna, Austria, May 2001.

[232] R. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3, pages 9-44, 1988.

[233] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction.* A Bradford Book. MIT Press, Cambridge, MA, 1998.

[234] J. Swevers, C. Gansemann, D. B. Tükel, J. De Schutter, and H. Van Brussel. Optimal robot excitation and identification. *IEEE Trans. on Robotics and Automation*, 13(5), pages 730-740, Oct. 1997.

[235] Manufacturing software, Tecnomatix. http://www.tecnomatic.com/, 2001.

[236] G. Thater und A. Behrens. Minimierung der Poseabweichung von Industrierobotern durch eine fehlerkompensierende Modellierung und Simulation der kinematischen Kette. In *Fachtagung Intelligente Steuerung und Regelung von Robotern*, VDI Bericht Nr. 1094, Seiten 501-510, Langen, Nov. 1993.

[237] M. Thümmel, M. Otter, and J. Bals. Control of robots with elastic joints based on automatic generation of inverse dynamics models. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 925–930, Maui, Hawaii, Oct./Nov. 2001.

[238] H. Tolle. Lernende Regelungen für die Automatisierungstechnik. In *GMA Aussprachetag „Wissensverarbeitung in der Automatisierungstechnik“*, VDI Bericht Nr. 897, Seiten 27-54, Langen, Juni 1991.

[239] K. Tomiyama, T. Furuta, and T. Noguchi. Three hybrid control schemes consisting of neural network and adaptive controllers for robot manipulators. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 1987–1992, Victoria, B. C., Canada, Oct. 1998.

[240] C. Torras. Robot neurocontrol: An overview. In *Int. Conf. on Artificial Neural Networks ICANN '95*, volume 1, Paris, France, pages 439-448, Oct. 1995.

[241] T. Tsuji, H. Akamatsu, and M. Kaneko. Non-contact impedance control for redundant manipulators using visual information. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2571–2576, Albuquerque, New Mexico, April 1997.

[242] S. Türk. *Zur Modellierung der Dynamik von Robotern mit rotatorischen Gelenken.* Fortschritt-Berichte, Reihe 8, Nr. 211. VDI-Verlag, Düsseldorf, 1990.

[243] H. Unbehauen. *Regelungstechnik 3 - Identifikation, Adaption, Optimierung.* Vieweg und Sohn, Braunschweig, 5. Auflage, 1995.

[244] T. Valency and M. Zacksenhouse. Instantaneous model impedance control of robots. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 757–762, Takamatsu, Japan, Oct./Nov. 2000.

[245] A. van den Bossche, F. Geuens, and U. Wienand. Rodym 6D, GCS100 - Messsysteme zur Analyse von Robotern. In *Fachtagung Robotik 2000*, VDI Bericht Nr. 1552, Seiten 133-138, Berlin, Juni 2000.

[246] P. M. J. van den Hof, P. S. C. Heuberger, and J. Bokor. System identification with generalized orthonormal basis functions. *Automatica*, 31(12), pages 1821-1834, 1995.

[247] P. van der Smagt. Feed-forward networks, 1998. http://www.robotic.dlr.de/Smagt/research/feed-forward/.

[248] P. van der Smagt and F. Groen. Visual feedback in motion. In O. Omidvar and P. van der Smagt, editors, *Neural Systems for Robotics*, pages 37–73. Academic Press, 1997.

[249] VDI 2861 (Technische Regel): Montage und Handhabungstechnik; Kenngrößen für Industrieroboter. Beuth Verlag, Berlin, Wien, Zürich, Ausgabe 1988-05.

[250] W. Verdonck, J. Swevers, X. Chenut, and J. C. Samin. Combining internal and external robot models to improve model parameter estimation. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2846–2851, Seoul, Korea, May 2001.

[251] M. Vincze. Dynamics and system performance of visual servoing. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 644–549, San Francisco, CA, April 2000.

[252] M. Vincze and G. D. Hager, editors. *Robust Vision for Vision-Based Control of Motion*. IEEE Press, 2000.

[253] C. J. C. H. Watkins. *Learning with Delayed Reward*. PhD thesis, Cambridge University, Psychology Department, 1989.

[254] T. Wengerek. *Reinforcment-Lernen in der Robotik*. Dissertationen zur künstlichen Intelligenz, Band 119. infix, Sankt Augustin, 1996.

[255] P. J. Werbos. Backpropagation through time: What is it and how to do it. *Proc. of the IEEE*, 78(10), pages 1550-1560, Oct. 1990.

[256] L. Whitcomb, S. Arimoto, T. Naniwa, and F. Ozaki. Experiments in adaptive model-based force control. In *IEEE Int. Conf. on Robotics and Automation*, pages 1846–1853, Nagoya, Japan, May 1995.

[257] D. E. Whitney and J. L. Nevins. What is the remote center compliance (RCC) and what can it do? In *9th Int. Symp. on Industrial Robots*, pages 135–152, Washingtion D. C., 1979.

[258] J. Wille. Untersuchungen zur HESSE-Matrix in Feedforward-Netzen. In *Beiträge zum Cottbuser Workshop 'Aspekte Neuronalen Lernens' CoWAN'96*, Seiten 41-52, TU Cottbus, Sept./Okt. 1996.

[259] J. Wille and T. Kolb. On improved learning algorithms with adaptive parameter regulation in feedforward nets. In *IEEE Int. Conf. on Neural Networks ICNN'95*, pages 115–121, Perth, Austalia, 1995.

[260] H. Wörn und E. Postenrieder. PFC-Regelung. In *Fortschrittliche Robotersteuerungstechnik (Hrsg. D. Schmid)*, Fachberichte Messen, Steuern, Regeln, Band 24, Seiten 178-191. Springer, 1991.

[261] H. Wörn, G. Rekewitz, D. Schmid und E. Michalak. Geschwindigkeitsgesteuerte Achsregelung. In *Fortschrittliche Robotersteuerungstechnik (Hrsg. D. Schmid)*, Fachberichte Messen, Steuern, Regeln, Band 24, Seiten 192-199. Springer, 1991.

[262] T. Yamada. Remarks on hybrid neural network controller using different convergence speeds. In *IEEE Int. Conf. on Robotics and Automation*, pages 562–568, Nagoya, Japan, May 1995.

[263] J. H. Yang, F. L. Lian, and L. C. Fu. Nonlinear adaptive control for flexible-link manipulators. *IEEE Trans. on Robotics and Automation*, 13(1), pages 140-148, Feb. 1997.

[264] Y. Yokokohji, Y. Sugawara, and T. Yoshikawa. Accurate image overlay on head-mounted displays using vision and accelerometers. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3243–3248, Detroit, Michigan, May 1999.

[265] B. K. Yoo and W. C. Ham. Adaptive control of robot manipulators using fuzzy compensator, part I and part II. In *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, pages 35–40, 52–57, Kyongju, Korea, Oct. 1999.

[266] T. Yoshikawa. Force control of robot manipulators. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 220–226, San Francisco, CA, April 2000.

[267] W.-S. Yu and G.-C. Wang. Adaptive control design using delayed dynamical neural networks for a class of nonlinear systems. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 3447–3452, Seoul, Korea, May 2001.

[268] U. Zachmann and K. Berns. Reinforcement-Learning bei der Steuerung eines autonomen mobilen Roboters. *Robotersysteme*, 7, Seiten 223-230, 1991.

[269] G. Zeng and A. Hemami. An adaptive control strategy for robotic cutting. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 22–27, Albuquerque, New Mexico, April 1997.

[270] J. Zhang, R. Schmidt, and A. Knoll. Appearance-based visual learning in a neuro-fuzzy model for fine-positioning of manipulators. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 1170–1175, Detroit, Michigan, May 1999.

[271] H. Zhuang. Hand / eye calibration for electronic assembly robots. *IEEE Trans. on Robotics and Automation*, 14(4), pages 612-616, Aug. 1998.

[272] H. Zhuang and Y. Meng. Using a scale: Self-calibration of a robot system with factor method. In *Proc. IEEE Int. Conference on Robotics and Automation*, pages 2797–2803, Seoul, Korea, May 2001.

[273] Z. Zografski and T. Durrani. Comparing predictions from neural networks and memory-based learning. In *Int. Conf. on Artificial Neural Networks ICANN '95*, volume 2, Paris, France, pages 221-226, Oct. 1995.

Anhang A

Kalman Filterung

Das Kalman Filter geht zurück auf eine Veröffentlichung von R. E. Kalman [119] und wurde seitdem in vielen Veröffentlichungen beschrieben. Da in verschiedenen Kapiteln der Arbeit darauf Bezug genommen wird, soll jetzt ein kurzer Überblick folgen, der im Gegensatz zu Lehrbüchern auch die in dieser Arbeit realisierten Algorithmen enthält. Eine ausführlichere Beschreibung ist in [149] enthalten. Dabei soll zunächst der Ansatz erläutert werden, bevor die einzelnen Algorithmen folgen.

Bei dem Kalman Algorithmus wird von zwei linearen Gleichungen ausgegangen, der Dynamikgleichung

$$\theta(k+1) = \mathbf{\Phi}(k) \cdot \theta(k) + \gamma(k) + \mathbf{v}(\zeta, k) \tag{A.1}$$

und der Messgleichung

$$y(k) = \psi^T(k) \cdot \theta(k) + w(\zeta, k). \tag{A.2}$$

Dabei sind $\mathbf{v}$ und w mittelwertfreie weiße Störprozesse, die sowohl untereinander als auch gegenüber den n zu schätzenden Parametern θ an vorangegangenen Abtastzeitpunkten unkorreliert sind. Dagegen ist eine Korrelation der Schätzwerte mit den Störungen vergangener Zeitpunkte nach Gleichung (A.1) vorhanden:

$$E\{\mathbf{v}(\zeta, k_1) \cdot \mathbf{v}^T(\zeta, k_2)\} = \mathbf{0} \qquad \forall k_1 \neq k_2 \tag{A.3}$$

$$E\{w(\zeta, k_1) \cdot w(\zeta, k_2)\} = 0 \qquad \forall k_1 \neq k_2 \tag{A.4}$$

$$E\{\mathbf{v}(\zeta, k_1) \cdot w(\zeta, k_2)\} = \mathbf{0} \qquad \forall k_1, k_2 \tag{A.5}$$

$$E\{\mathbf{v}(\zeta, k_1) \cdot \theta^T(k_2)\} = \mathbf{0} \qquad \forall k_1 \leq k_2 \tag{A.6}$$

$$E\{w(\zeta, k_1) \cdot \theta(k_2)\} = \mathbf{0} \qquad \forall k_1, k_2 \tag{A.7}$$

Dabei steht $E\{..\}$ für den Erwartungswert. Die Varianzen der Rauschprozesse sind durch

$$E\{\mathbf{v}(\zeta, k) \cdot \mathbf{v}^T(\zeta, k)\} = \mathbf{Q}(k) \tag{A.8}$$

und

$$E\{w(\zeta, k) \cdot w(\zeta, k)\} = \sigma^2(k) \tag{A.9}$$

abgeschätzt. $\Phi(k)$, $\gamma(k)$, $\psi(k)$ und $y(k)$ sind zum Zeitpunkt k ebenfalls bekannt.

Das Kalman Filter bestimmt einen Schätzvektor $\hat{\theta}$ für den Vektor der Unbekannten θ, wobei der quadratischer Fehler

$$E\{(\hat{\theta} - \theta)^T \cdot (\hat{\theta} - \theta)\} \tag{A.10}$$

minimal ist.

Dabei geht man von Anfangswerten

$$\hat{\theta}(0) = \mathbf{0} \tag{A.11}$$

aus, wenn man keine besseren a priori Informationen hat. Dies entspricht der Annahme von mittelwertfreien Unbekannten.

Außerdem wird eine Kovarianzmatrix

$$\mathbf{P}(k) = E\{(\hat{\theta}(k) - \theta(k)) \cdot (\hat{\theta}(k) - \theta(k))^T\} \tag{A.12}$$

definiert, deren Anfangswert $\mathbf{P}(0)$ ebenfalls vorgegeben wird. Bei Annahme von Gleichung (A.11) ist dies

$$\mathbf{P}(0) = E\{\theta(0) \cdot \theta^T(0)\}. \tag{A.13}$$

$\mathbf{P}(0)$ wird meist als Diagonalmatrix mit den Elementen p_{0i} angesetzt. Später kann man aus Gleichung (A.12) die Qualität der Schätzung ablesen, da die Elemente der Diagonale von $\mathbf{P}(k)$ gleich den geschätzten quadratischen Parameterfehlern sind. Dies setzt voraus, dass die Wahl von $\mathbf{P}(0)$, $\mathbf{Q}(k)$ und $\sigma^2(k)$ richtig erfolgt. Es zeigt sich aber, dass der Algorithmus gutartig gegenüber fehlerhaften Vorgaben reagiert, sodass die Angabe einer Größenordnung i. A. ausreicht.

Die folgenden Abschnitte beschreiben nun die einzelnen Algorithmen. Zunächst wird das „normale“ Kalman Filter ausgeführt. Dann folgt eine Spezialversion, die unter bestimmten Bedingungen große Rechenzeitvorteile bringt. Beide Algorithmen gelten für lineare Systeme. Im Anhang A.3 folgt dann die Erweiterung auf nichtlineare Systeme, die im Anhang B zum Training Neuronaler Netze verwendet wird.

A.1 Rekursives Kalman Filter

Für das Kalman Filter gibt es einen rekursiven und einen nichtrekursiven Algorithmus, wobei Letzterer nicht betrachtet wird, da er keine Zeitvarianz erlaubt.

Außerdem sollen die Gleichungen für den Sonderfall $\mathbf{\Phi}(k) = \mathbf{I}$, $\gamma(k) = \mathbf{0}$ betrachtet werden, also für eine rein stochastische Dynamik.

Das entspricht der Identifikation von Parametern einer linearen Gleichung, wie sie in Abschnitt 5.1 zum Training des Modells angesetzt wurde. Außerdem verwendet die unterste Adaptionsebene (Abschnitt 5.4) diesen Algorithmus zur Bestimmung der Parameter.

Die Gleichungen des rekursiven Kalman Filters lauten dann:

$$\mathbf{P}^*(k+1) = \mathbf{P}(k) + \mathbf{Q} \tag{A.14}$$

$$\begin{aligned} \hat{\theta}(k+1) &= \hat{\theta}(k) + \mathbf{P}^*(k+1) \cdot \psi(k+1) \\ &\cdot \quad (\psi^T(k+1) \cdot \mathbf{P}^*(k+1) \cdot \psi(k+1) + \sigma^2(k+1))^{-1} \\ &\cdot \quad (y(k+1) - \psi^T(k+1) \cdot \hat{\theta}(k)) \end{aligned} \tag{A.15}$$

$$\begin{aligned} \mathbf{P}(k+1) &= \mathbf{P}^*(k+1) - \mathbf{P}^*(k+1) \cdot \psi(k+1) \cdot \psi^T(k+1) \cdot \mathbf{P}^*(k+1) \\ &\cdot \quad (\psi(k+1)^T \cdot \mathbf{P}^*(k+1) \cdot \psi(k+1) + \sigma^2(k+1))^{-1}. \end{aligned} \tag{A.16}$$

Gleichung (A.14) beschreibt dabei die in diesem Fall sehr einfache Prädiktion von den Messwerten des Schrittes k auf den Schritt $k+1$. Die Gleichungen (A.15) und (A.16) nehmen dann die Korrektur aufgrund des Messwertes zum Zeitpunkt $k+1$ vor. Dabei bezeichnet man

$$\mathbf{K}(k+1) = \mathbf{P}^*(k+1) \cdot \psi(k+1) \cdot (\psi(k+1)^T \cdot \mathbf{P}^*(k+1) \cdot \psi(k+1) + \sigma^2(k+1))^{-1} \tag{A.17}$$

auch als Kalman Verstärkung.

Die zu invertierende Matrix ist ein Skalar, der wegen der quadratischen Form und der Addition von $\sigma^2 > 0$ immer positiv ist.

Zur schnelleren Berechnung empfiehlt sich die Zwischenspeicherung von mehrfach vorkommenden Ausdrücken. Außerdem reicht es, von der symmetrischen Matrix $\mathbf{P}$ nur eine Dreiecksmatrix zu berechnen. Dies ist in [149] im Detail beschrieben. Der Rechenaufwand ist damit proportional zur Zahl der Messgleichungen N und zum Quadrat der Zahl der Unbekannten n. Der entwickelte Algorithmus benötigt je Messgleichung $3/2n^2 + 5n$ Multiplikationen und Additionen.

Von den numerischen Eigenschaften kann man annehmen, dass es keine Probleme gibt, wenn die zu invertierende Größe einigermaßen genau berechnet wird.

Das setzt u. a. voraus, dass $\psi^T \cdot \mathbf{P}^* \cdot \psi$ und σ^2 nicht völlig unterschiedliche Größenordnungen haben. Eine Faustregel für die Rechnung mit einfacher Genauigkeit (4 byte Zahlendarstellung) ist dabei

$$\psi^T \cdot \mathbf{P}^* \cdot \psi \leq 10^6 \cdot \sigma^2. \tag{A.18}$$

Dies bedeutet

$$\sum_{i=1}^{n} \sum_{j=1}^{n} \psi_i \cdot p_{ij}^* \cdot \psi_j \leq 10^6 \cdot \sigma^2. \tag{A.19}$$

Dabei müssen die Elemente der Messvektoren für den ungünstigsten Fall abgeschätzt werden, da andernfalls nicht garantiert werden kann, dass $\mathbf{P}$ positiv definit ist, was schwerwiegende Folgen für den Algorithmus hat. Es ergibt sich damit bei Vorgabe von $\mathbf{P}^*$ ein Minimalwert für σ^2, der nicht unterschritten werden sollte, um die Konvergenz nicht zu gefährden.

Bei einer positiv definiten Matrix $\mathbf{P}^*$ gilt

$$p_{ii}^* \cdot p_{jj}^* \geq p_{ij}^{*2} \qquad \forall i, j \tag{A.20}$$

Daraus folgt

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{n} \psi_i \cdot p_{ij}^* \cdot \psi_j &\leq \max_i \psi_i^2 \sum_{i=1}^{n} \sum_{j=1}^{n} p_{ij}^* \\ &\leq \max_i \psi_i^2 \sum_{i=1}^{n} \sqrt{p_{ii}^*} \cdot \sum_{j=1}^{n} \sqrt{p_{jj}^*} \\ &\leq \max_i \psi_i^2 \left(\sum_{i=1}^{n} \sqrt{p_{ii}^*} \right)^2 \end{aligned} \tag{A.21}$$

Deshalb wird ein Minimalwert für σ festgelegt:

$$\sigma^2 \geq 10^{-6} \cdot \left(\sum_{i=1}^{n} \sqrt{p_{ii}^*} \right)^2 \cdot \max_i \psi_i^2 \tag{A.22}$$

Es kann festgestellt werden, dass

$$\left(\sum_{i=1}^{n}\sqrt{p_{ii}^*}\right)^2 \geq \sum_{i=1}^{n} p_{ii}^*, \tag{A.23}$$

dass es also nicht reicht, σ^2 durch $\sum_{i=1}^{n} p_{ii}^*$ abzuschätzen.

Außerdem kann man feststellen, dass die Schätzung ungenau wird, wenn $\psi_i \cdot p_{ii}^* \cdot \psi_i$ bei den verschiedenen Elementen unterschiedliche Größenordnungen hat. So werden Summanden als null interpretiert, wenn sie um den Faktor 10^{-7} kleiner sind als der Gesamtausdruck. Es kann also angeraten sein, die Unbekannten geeignet zu skalieren.

A.2 Inverser Kalman Algorithmus für große Gleichungssysteme

Der Aufwand zur Korrektur ist beim normalen Kalman Algorithmus proportional n^2, wobei n die Zahl der Unbekannten ist. Dies gilt, wenn der Messvektor ψ auch n von null verschiedene Elemente hat. Es gibt aber Aufgaben, bei denen der Messvektor größtenteils aus Nullen besteht. In solchen Fällen kann der Rechenaufwand reduziert werden, ohne dass sich an den Eigenschaften der Schätzung etwas ändert. Das wird in diesem Abschnitt beschrieben.

Angewandt wird der Algorithmus in Abschnitt 5.2 zur Bestimmung von Stellfolgen bzw. Teiltrajektorien aus den Werten der Gewichtsfunktion und der beobachteten Regeldifferenzen.

Der oben beschriebene Algorithmus verbessert rekursiv die Schätzwerte aufgrund neuer Messgleichungen mit vektoriellem ψ und skalarem y. Der ohne Zeitvarianz gleichwertige nichtrekursive Algorithmus fasst alle N Messgleichungen in einer Vektorgleichung mit der Matrix $\mathbf{\Psi}$, bestehend aus den Spaltenvektoren $\psi(k)$, und einem Vektor $\mathbf{y}$ mit den Elementen $y(k)$ zusammen. Den Algorithmus kann man dann auch geschlossen hinschreiben [169]:

$$\hat{\theta}(N) = (\mathbf{\Psi} \cdot \mathbf{R}^{-1} \cdot \mathbf{\Psi}^T + \mathbf{P}_0^{-1})^{-1} \cdot \mathbf{\Psi} \cdot \mathbf{R}^{-1} \cdot \mathbf{y} \tag{A.24}$$

$$\mathbf{P}(N) = (\mathbf{\Psi} \cdot \mathbf{R^{-1}} \cdot \mathbf{\Psi}^T + \mathbf{P_0^{-1}})^{\mathbf{-1}} \tag{A.25}$$

Dabei ist R die Diagonalmatrix der Elemente

$$r_{kk} = \sigma^2(k). \tag{A.26}$$

Aus diesen Gleichungen kann man eine rekursive Rechenvorschrift herleiten für

$$\mathbf{P'} = \mathbf{P}^{-1}(N) = \mathbf{P_0^{-1}} + \mathbf{\Psi} \cdot \mathbf{R^{-1}} \cdot \mathbf{\Psi}^T = \mathbf{P_0^{-1}} + \sum_{k=1}^{N} \frac{\psi(k) \cdot \psi^T(k)}{\sigma^2(k)} \tag{A.27}$$

und

$$\theta' = \mathbf{P'} \cdot \hat{\theta} = \mathbf{\Psi} \cdot \mathbf{R^{-1}} \cdot \mathbf{y} = \sum_{k=1}^{N} \frac{\psi(k) \cdot y(k)}{\sigma^2(k)}. \tag{A.28}$$

Sie lautet:

$$\mathbf{P'}(k) = \mathbf{P'}(k-1) + \frac{\psi(k) \cdot \psi^T(k)}{\sigma^2(k)} \tag{A.29}$$

$$\theta'(k) = \theta'(k-1) + \frac{\psi(k) \cdot y(k)}{\sigma^2(k)} \tag{A.30}$$

Die Gleichungen sind wesentlich einfacher als die entsprechenden Gleichungen des normalen rekursiven Kalman Filters (Gleichungen (A.15) und (A.16)). Zur Berechnung des Schätzvektors $\hat{\theta}$ selbst muss allerdings noch das lineare Gleichungssystem (A.28) gelöst werden.

Das beschränkt die Anwendung des inversen Kalman Filters auf Schätzprobleme, bei denen die Vektoren ψ spärlich besetzt sind und auf Aufgaben, bei denen wesentlich mehr Messgleichungen als Schätzwerte vorliegen ($N \gg n$).

Nachteil des Algorithmus ist es, dass keine Prädiktion möglich ist, nicht einmal die Addition von $\mathbf{Q}$ wegen einer stochastischen Änderung des Schätzvektors, wie in Gleichung (A.14).

Da $\mathbf{P}(N)$ a priori nicht vorliegt, sind als Anfangswerte auch nur $\hat{\theta}_i(0) = 0$ möglich, was aber kaum stört, da man immer die Differenz zu anderen Anfangswerten schätzen kann.

Die Lösung des Gleichungssystems erfolgt normalerweise durch Choleski-Zerlegung von $\mathbf{P}'$ in zwei gleiche Dreiecksmatrizen ($\mathbf{P}' = \mathbf{U}' \cdot \mathbf{U}'^T$, siehe z. B. [85]) oder zwei gleiche Dreiecksmatrizen und eine Diagonalmatrix ($\mathbf{P}' = \mathbf{U}' \cdot \mathbf{D}' \cdot \mathbf{U}'^T$). Dazu wird nun ein modifizierter Algorithmus mit zwei unterschiedlichen Dreiecksmatrizen vorgestellt, der die Wurzelbildung vermeidet und die Berechnung der Kovarianzmatrix für eine begrenzte Zahl von Elementen ermöglicht (ausführlich in [146]). Der Ansatz lautet:

$$\mathbf{P}' = \mathbf{U}'_1 \cdot \mathbf{U}'_2 \tag{A.31}$$

Da $\mathbf{P}'$ wegen der spärlichen Besetzung der Vektoren ψ mit je $m = n_g$ von null verschiedenen Koeffizienten eine Bandmatrix mit $2m-1$ Elementen je Zeile ist, haben auch die Dreiecksmatrizen Bandcharakter ([220]):

$$\mathbf{U}'_2 = \begin{bmatrix} u'_{11} & & & & \mathbf{0} \\ \cdots & \cdots & & & \\ u'_{m1} & & \cdots & & \\ & \cdots & & \cdots & \\ \mathbf{0} & & u'_{n,n-m+1} & \cdots & u'_{nn} \end{bmatrix} \tag{A.32}$$

$$\mathbf{U}'_1 = \begin{bmatrix} \frac{u'_{11}}{u'_{11}} & \cdots & \frac{u'_{m1}}{u'_{mm}} & & \mathbf{0} \\ & \cdots & & \cdots & \\ & & \cdots & & \frac{u'_{n,n-m+1}}{u'_{nn}} \\ & & & \cdots & \cdots \\ \mathbf{0} & & & & \frac{u'_{nn}}{u'_{nn}} \end{bmatrix} \tag{A.33}$$

Man berechnet

$$u'_{ij} = p'_{ij} - \sum_{k=i+1}^{min(j+m-1,n)} \frac{u'_{ki} \cdot u'_{kj}}{u'_{kk}}. \tag{A.34}$$

Dabei können für die u'_{ij} die gleichen Speicherplätze wie für die p'_{ij} benutzt

werden.

Die Lösung des Gleichungssystems ergibt sich damit in zwei Schritten zu

$$\theta''_i = \theta'_i - \sum_{k=i+1}^{min(i+m-1,n)} \frac{u'_{ki}}{u'_{kk}} \cdot \theta''_k \tag{A.35}$$

und

$$\hat{\theta}_i = \frac{1}{u'_{ii}} \cdot \left[\theta''_i - \sum_{k=max(i-m+1,1)}^{i-1} u'_{ik} \cdot \hat{\theta}_k \right]. \tag{A.36}$$

Dabei können $\hat{\theta}$, θ' und θ'' auch den gleichen Speicherbereich benutzen. Zur Lösung des Gleichungssystems sind also etwa $n \cdot m^2/2 + 2 \cdot n \cdot m$ Operationen nötig.

Leider steigt diese Zahl stark an, wenn auch die Kovarianzmatrix $\mathbf{P}$ benötigt wird. Das liegt daran, dass $\mathbf{P}$ im Gegensatz zu $\mathbf{P}'$ keine Bandstruktur hat. Zur Bestimmung berechnet man zuerst die Dreiecksmatrix $\mathbf{U}_2$, die Inverse von $\mathbf{U}'_2$, die auch keine Bandstruktur hat. Mit der Bezeichung u_{ij} für die Elemente von $\mathbf{U}_2$ und δ_{ij} für die Impulsfunktion

$$\delta_{ij} = \begin{cases} 1 & \textit{für} \quad i = j \\ 0 & \textit{für} \quad i \neq j \end{cases} \tag{A.37}$$

ergibt sich

$$u_{ij} = \frac{1}{u'_{ii}} \cdot \left[\delta_{ij} - \sum_{k=max(j,i-m+1)}^{i-1} u'_{ik} \cdot u_{kj} \right]. \tag{A.38}$$

Die Berechnung der Kovarianzmatrix $\mathbf{P}$ erfolgt dann durch

$$p_{ij} = \sum_{k=1}^{j} \frac{u_{ik} \cdot u_{jk}}{u_{kk}}. \tag{A.39}$$

Dabei kann mit $\mathbf{P}$ der Speicherplatz von $\mathbf{U}_2$ überschrieben werden, wenn die

Elemente zeilenweise von unten nach oben und innerhalb der Zeile von rechts nach links bestimmt werden.

Die Berechnung in der angegebenen Reihenfolge erlaubt die Beschränkung auf die ersten l Zeilen bzw. Spalten von $\mathbf{P}$. Dadurch steigt die Rechenzeit gegenüber der reinen Lösung des Gleichungssystems nur um $l^2/2 \cdot m + l^2/2$ Operationen.

Durch diesen Algorithmus reduziert sich der Rechenaufwand zur Bestimmung von n Schätzwerten aufgrund von N Messgleichungen, bei denen jeweils m Elemente des Messvektors ψ von null verschieden sind, auf etwa $N \cdot (m^2/2+m)+ n \cdot m^2/2 + 2 \cdot n \cdot m$ Operationen. Bei zusätzlicher Abschätzung des Schätzfehlers für l Schätzwerte (Berechnung von $\mathbf{P}$ aus $\mathbf{P}'$) kommen $l^2/2 \cdot (m+1)$ Operationen dazu [146]. Dies ist unter Umständen erheblich weniger als die $(3/2 \cdot n^2 + 5 \cdot n) \cdot N$ Operationen des normalen rekursiven Algorithmus.

Bei der Anwendung in Abschnitt 6.2 gilt $N \approx n \approx 200$ und $m = 7$. Das ergibt zur Stellfolgenbestimmung $14 \cdot 10^3$ Operationen anstelle von $12 \cdot 10^6$, wobei zur Parameteradaption für die Vorsteuerung bei $l = 10$ allerdings noch etwa 400 Operationen hinzukommen.

A.3 Erweitertes Kalman Filter

Im Gegensatz zu den bisherigen Formen des Kalman Filters werden beim erweiterten (extended) Kalman Filter die Parameter einer analytisch gegebenen nichtlinearen Messgleichung

$$y(k) = \psi(\theta, k) \tag{A.40}$$

bestimmt. Dabei wird angenommen, dass die Messgleichungen „genau“ sind, dass also kein Rauschen mit einer Störung σ^2 existiert.

Diese Aufgabe entspricht dem Training eines Neuronalen Netzes, das im Anhang B näher beschrieben ist.

Die nichtlineare Abhängigkeit kann man linearisieren durch

$$y(k) = \psi_0(k) + \psi^T(k) \cdot \theta, \tag{A.41}$$

wobei im Gegensatz zum linearen Fall $\psi_0(k)$ und $\psi(k)$ nicht bekannt, sondern selbst Funktionen von θ sind und daher jeweils aus der gegebenen Funktion $\psi(\theta, k)$ berechnet werden müssen.

Aus diesem Grund ist keine direkte, sondern nur eine iterative Schätzung möglich. Die Kalman Gleichungen werden also nicht nur einmal für jede Messgleichung durchlaufen, sondern die Rekursion wird so oft wiederholt, bis der mittlere Fehler der Messgleichungen

$$\sqrt{\frac{1}{N}\sum_{k=1}^{N}(y(k)-\psi(\hat{\theta},k))^2} \tag{A.42}$$

eine vorgegebene Schranke unterschreitet. Dabei bezeichnet man das einmalige Abarbeiten aller Messgleichungen als Epoche.

$\psi_0(k)$ und $\psi(k)$ werden aufgrund des jeweils neuesten Schätzvektors $\hat{\theta}(k-1)$ gebildet (schrittweise Korrektur). Alternativ könnte man zu Beginn jeder Epoche aufgrund des dann gültigen $\hat{\theta}$ die $\psi_0(k)$ und $\psi(k)$ für alle Messgleichungen bestimmen (epochale Korrektur). Dies erweist sich aber meist als ungünstiger, insbesondere bei einer hohen Zahl an Messgleichungen.

Während die Gleichungen (A.14) und (A.16) unverändert gelten, muss bei Gleichung (A.15) $y(k)$ durch $y(k)-\psi_0(k)$ ersetzt werden. Dadurch ergibt sich

$$\begin{aligned}\hat{\theta}(k+1) &= \hat{\theta}(k)+\mathbf{K}(k+1)\cdot(y(k+1)-\psi_0(k+1)-\psi^T(k+1)\cdot\hat{\theta}(k)) \\ &= \hat{\theta}(k)+\mathbf{K}(k+1)\cdot(y(k+1)-\psi(\hat{\theta}(k),k+1))\end{aligned} \tag{A.43}$$

mit $\mathbf{K}(k+1)$ aus Gleichung (A.17).

Die Gleichungen (A.14), (A.16), (A.43) und (A.17) bilden das erweiterte Kalman Filter für den Fall $\mathbf{\Phi}=\mathbf{I}$ und $\gamma=\mathbf{0}$, der schon bei der Beschreibung des rekursiven Kalman Filters betrachtet wurde. σ^2 und $\mathbf{Q}$ sind dabei freie Parameter, die so gewählt werden, dass die Zahl der Epochen minimal wird.

Das beschreibt der folgende Anhang, in dem der Algorithmus auf das Training Neuronaler Netze angewendet wird.

Anhang B

Training von Neuronalen Netzen - EKFNet

Da die bekannten Algorithmen zum Training von Netzen nur sehr langsam konvergieren, wurde ein neues Verfahren entwickelt [151], um die Netze in Kapitel 6 schnell trainieren zu können.

Beim Training eines schichtenorientierten Neuronalen Netzes ohne Rückkopplung handelt es sich um die Bestimmung einer oder mehrerer nichtlinearer Funktionen. Man kann also das erweiterte Kalman Filter nach Anhang A.3 anwenden. Dieser Ansatz wurde in ähnlicher Form auch von anderen Autoren vorgeschlagen [87, 120, 226].

Die Gewichte des Neuronalen Netzes sind dabei die Schätzwerte der Messgleichung (A.40). Sie sollen so bestimmt werden, dass die Ausgänge des Netzes den Sollwerten möglichst nahe kommen. Dabei sind verschiedene Datensätze vorhanden, die aus Eingangsdaten und Sollwerten für $y(k)$ bestehen. Zur Anwendung des erweiterten Kalman Filters müssen die Eingangsdaten allerdings noch in die Werte $\psi(\theta, k)$ und $\psi(k)$ umgerechnet werden. Dabei wird die Jacobi-Matrix ψ^T in diesem Abschnitt mit $\mathbf{J}$ bezeichnet, da die Indizierung anders als bisher erfolgt. Aus diesem Grund werden auch die Elemente des Parametervektors θ nun $weight$ [1] genannt.

Es wird hier der Fall betrachtet, dass das Netz nur einen Ausgang hat. Vektorielle Messgleichungen wurden beim Kalman Filter nicht getrennt betrachtet, da es zumindest im linearen zeitinvarianten Fall keinen Unterschied macht, wenn

[1] Die üblichen Bezeichnungen w, i, und o für Gewichte, Eingänge und Ausgänge werden zur Vermeidung von Verwechslungen hier nicht verwendet.

man verschiedene skalare Messgleichungen

$$y(k,i) = \psi^T(k,i) \cdot \theta(k) \tag{B.1}$$

zu einer vektoriellen Gleichung

$$\mathbf{y}(k) = \begin{bmatrix} y(k,1) \\ y(k,2) \\ y(k,3) \\ \cdots \end{bmatrix} = \begin{bmatrix} \psi^T(k,1) \\ \psi^T(k,2) \\ \psi^T(k,3) \\ \cdots \end{bmatrix} \cdot \theta = \Psi^T(k) \cdot \theta \tag{B.2}$$

zusammenfasst. Durch die Annahme von Zeitvarianz oder Nichtlinearität ergeben sich zwar Differenzen zwischen skalarer und vektorieller Betrachtung, die Bestimmung der Gewichte wird aber nicht beeinträchtigt, wenn weitere Ausgänge als zusätzliche Messgleichungen mit geänderten ψ_0 und ψ interpretiert werden.

Für einen festen Zeitpunkt und einen Ausgang hat die Jacobi-Matrix nur eine Zeile. Deren Bestimmung soll nun erfolgen. Dabei wird zur Vorstellung der Notation auch die Berechnung der Ausgänge aufgrund der Eingangsdaten und der Gewichte dokumentiert.

Für jedes Neuron i_1 in jeder Schicht i (außer der Eingangsschicht) ergibt sich die Aktivität

$$out(i,i_1) = \frac{1}{1+e^{-net(i,i_1)}} \tag{B.3}$$

mit

$$net(i,i_1) = weight(i,i_1,0) + \sum_{i_2=1}^{l(i-1)} weight(i,i_1,i_2) \cdot out(i-1,i_2). \tag{B.4}$$

Dabei ist $l(i-1)$ die Zahl der Neuronen der vorhergehenden Schicht. Bei der ersten verdeckten Schicht ist $out(i-1,i_2)$ durch den Eingang $in(i_2)$ zu ersetzen. $weight(i,i_1,i_2)$ ist das Gewicht, das Neuron i_2 in Schicht $i-1$ mit Neuron i_1 in Schicht i verbindet. $weight(i,i_1,0)$ ist dagegen die Schwelle von Neuoron i_1 in

Schicht i. Der Ausgang wird also *vorwärts* durch das Netz berechnet. Für das betrachtete Neuron i_{out} der Ausgangsschicht n_l gilt

$$\psi(\theta) = out(n_l, i_{out}). \tag{B.5}$$

Zur Berechnung der Jacobi-Matrix muss zuerst die Ableitung der Fermi Funktion (Gleichung (B.3)) bestimmt werden. Sie lautet:

$$\frac{\partial out(i, i_1)}{\partial net(i, i_1)} = out(i, i_1) - out(i, i_1) \cdot out(i, i_1) \tag{B.6}$$

Die Jacobi-Matrix kann nun für den Ausgang i_{out} hergeleitet werden. Dabei verwendet man die Kettenregel

$$J(i, i_1, i_2) = \frac{\partial out(n_l, i_{out})}{\partial weight(i, i_1, i_2)} = \frac{\partial out(n_l, i_{out})}{\partial net(n_l, i_{out})} \cdot \frac{\partial net(n_l, i_{out})}{\partial weight(i, i_1, i_2)}. \tag{B.7}$$

Für $i = n_l$ und $i_1 = i_{out}$ ist

$$\frac{\partial net(n_l, i_{out})}{\partial weight(i, i_1, i_2)} = out(i-1, i_2). \tag{B.8}$$

Für die übrigen Schichten muss man die Kettenregel mehrfach anwenden. Im Allgemeinen verwendet man die Hilfsgröße δ, die *rückwärts* im Netz berechnet wird. Sie ist hier etwas anders als bei dem herkömmlichen Back-Propagation Algorithmus definiert. Für die Ausgangsschicht ergibt sich damit:

$$\delta(n_l, i_1) = \begin{cases} 0 & für\, i_1 \neq i_{out} \\ out(n_l, i_1) - out(n_l, i_1) \cdot out(n_l, i_1) & für\, i_1 = i_{out} \end{cases} \tag{B.9}$$

Für die übrigen Schichten ergibt sich:

$$\delta(i, i_1) = \sum_{i_2=1}^{l(i+1)} \delta(i+1, i_2) \cdot weight(i+1, i_2, i_1) \cdot (out(i, i_1) - out(i, i_1) \cdot out(i, i_1)) \tag{B.10}$$

Daraus lassen sich die Elemente der Jacobi-Matrix bestimmen:

$$J(i, i_1, 0) = \delta(i, i_1) \tag{B.11}$$

$$J(i, i_1, i_2) = \delta(i, i_1) \cdot out(i-1, i_2) \tag{B.12}$$

Damit sind die Gleichungen zur Bestimmung der Gewichte durch ein erweitertes Kalman Filter beschrieben. Zur Rechenzeit lässt sich sagen, dass die Bestimmung des Ausgangswertes und der Jacobi-Matrix im Vergleich zum Kalman Filter schnell geht, da bei Letzterem der Aufwand quadratisch mit der Zahl der Gewichte steigt (siehe Abschätzung in Anhang A.1).

Die Lernparameter des erweiterten Kalman Filters lassen sich allgemein angeben, da zum Training eines Neuronalen Netzes die Trainingsdaten normalisiert sind. So liegen die Netzeingänge in der Größenordnung

$$in(i_{in}) \subseteq [0, 1], \tag{B.13}$$

während die Netzausgänge die Bedingung

$$0.05 < out(n_l, i_{out}) < 0.95 \tag{B.14}$$

erfüllen müssen.

Die Erwartungswerte der Schätzwerte lassen sich dann durch

$$\mathbf{P}_0 = \mathbf{I} \tag{B.15}$$

vorgeben, während σ^2 und $\mathbf{Q}$ am besten angepasst werden.

Wie oben schon erwähnt, hat σ^2 beim Neuronalen Netz keine physikalische Bedeutung, da die Sollwerte als fehlerfrei gelten. Zur Vermeidung einer starken Korrektur am Anfang des Trainings empfiehlt sich ein hoher Wert, der mit fortschreitender Güte der Abbildung von Epoche zu Epoche langsam sinkt. Bewährt hat sich

$$\sigma^2 = 0.1 \cdot \frac{1}{N} \sum_{k=1}^{N} (y(k) - \psi(\theta, k))^2. \tag{B.16}$$

Zur Vermeidung von numerischen Problemen darf aber der Minimalwert nach Gleichung (A.22) nicht unterschritten werden.

Für $\mathbf{Q}$ ist trotz zeitinvariantem Netz auch ein von null verschiedener Wert sinnvoll. Der Grund dafür liegt in der Nichtlinearität, die im Kalman Filter eigentlich nicht vorgesehen ist und daher nur als Zeitinvarianz modelliert werden kann. Und zwar ist die Zeitvarianz umso größer, je mehr sich die Jacobi-Matrix ändert. Heuristisch wird deshalb

$$\mathbf{Q} = \frac{q}{n} \sum_{i=1}^{n} (\theta_i - \theta_{alt,i})^2 \cdot \mathbf{I} \tag{B.17}$$

angesetzt, wobei sich $q = 0.01$ bewährt hat. Dieses $\mathbf{Q}$ wird nur nach jeder Epoche zur Kovarianzmatrix $\mathbf{P}$ addiert, also jedesmal, wenn alle Messgleichungen abgearbeitet worden sind, da zwischendurch größere Schwankungen der Schätzwerte möglich sind.

Anhang C

Testbahnen zur Beurteilung der Bahngenauigkeit

Alle Testbahnen sind so definiert, dass die Stellgrößenbegrenzungen bei korrekter Ausführung nicht erreicht werden. Die optimalen Kommandos verlassen also nicht den Bereich der erlaubten Gelenkwinkel, -geschwindigkeiten und -beschleunigungen. Damit ist angenommen, dass die Bahnplanung eine realisierbare Bahn generiert hat, bei der diese Grenzen eingehalten werden.

Die Bahnen decken den gesamten Arbeitsraum des Roboters ab. Sie setzen sich zwar aus ebenen Teilbahnen zusammen, es kann aber davon ausgegangen werden, dass das Verhalten bei räumlichen Bahnen ähnlich ist, da im Achsraum bei diesen Bewegungen keine Ebene ausgezeichnet ist. Eine Ausnahme bildet lediglich Achse 4, deren Wert bei den Testbahnen mit senkrechtem Endeffektor konstant gehalten wird. Dadurch ist die Orientierung des Greifers während vieler Bahnen unverändert senkrecht nach oben bzw. unten (Ein waagrechter Greifer wäre aufgrund des hohen im Handgelenk wirkenden Moments beim Manutec Roboter nicht dauerhaft zulässig.). Es ist davon auszugehen, dass Aussagen, die für Translationen gelten, auch für Rotationen gültig sind, da es im Achsraum keine Unterscheidung zwischen diesen Arten der Bewegung gibt. Insofern enthält die Wahl der Testbahnen, abgesehen von der unbewegten 4. Achse, keinerlei Einschränkungen.

Die einzelnen Testbahnen sind in den folgenden Abbildungen zusammengefasst.

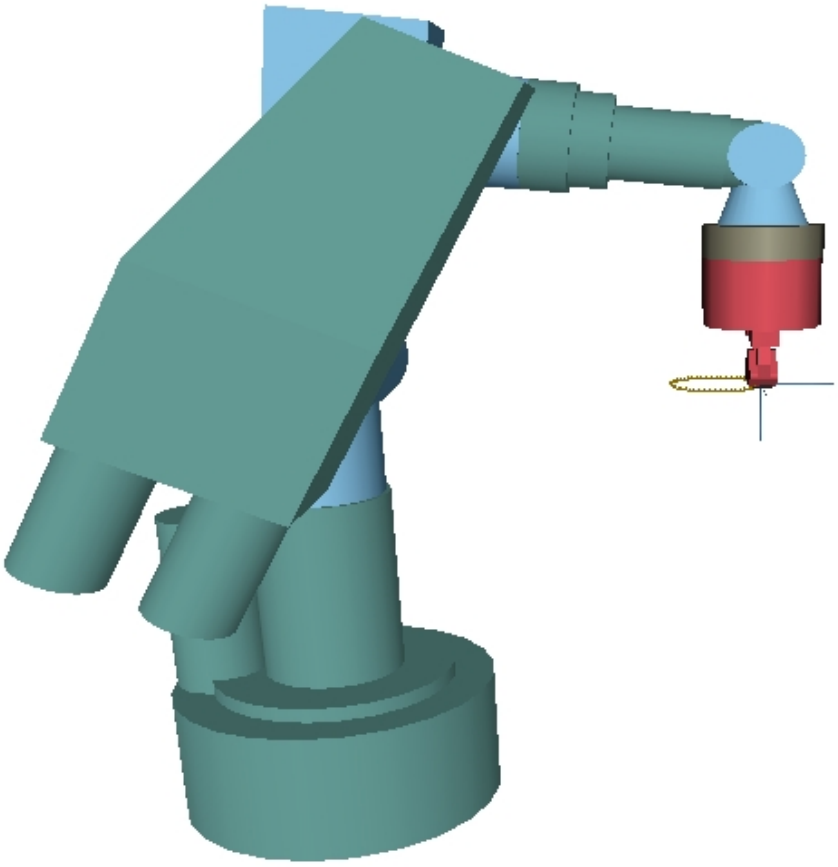


Abbildung C.1: Darstellung der Endeffektorpositionen und der Roboterstellung des Manutec r2 am Ende von Bahn 1.

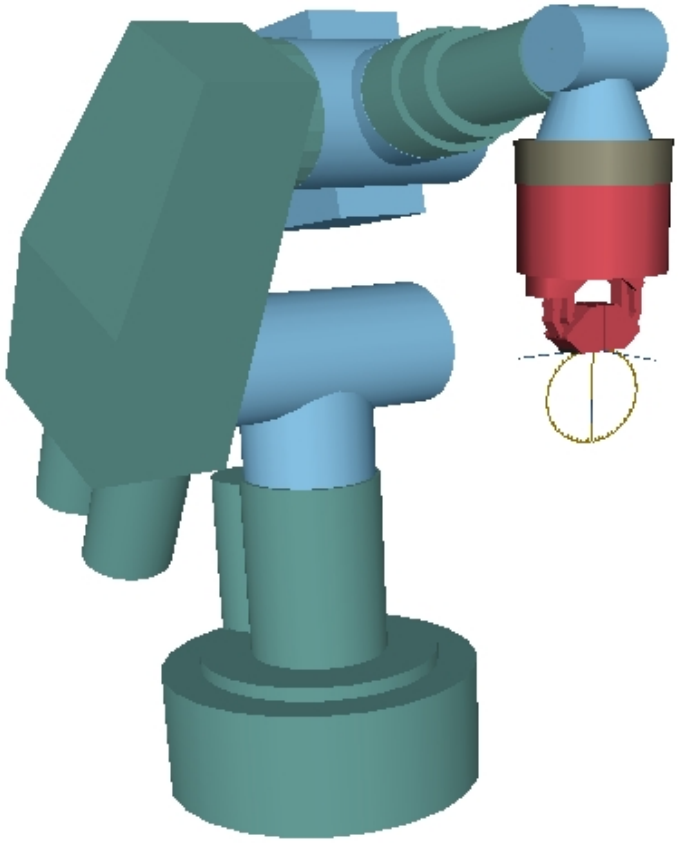


Abbildung C.2: Darstellung der Endeffektorpositionen bei Bahn 2. Am Ende wurde der Endeffektor senkrecht nach oben verschoben, weil er sonst die Bahn verdeckt.

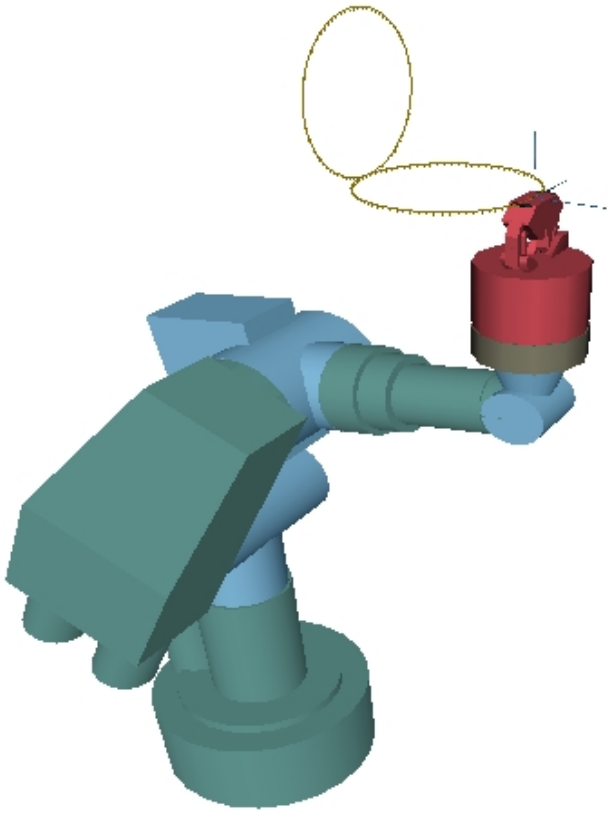


Abbildung C.3: Darstellung der Endeffektorpositionen und der Roboterstellung des Manutec r2 am Ende von Bahn 3.

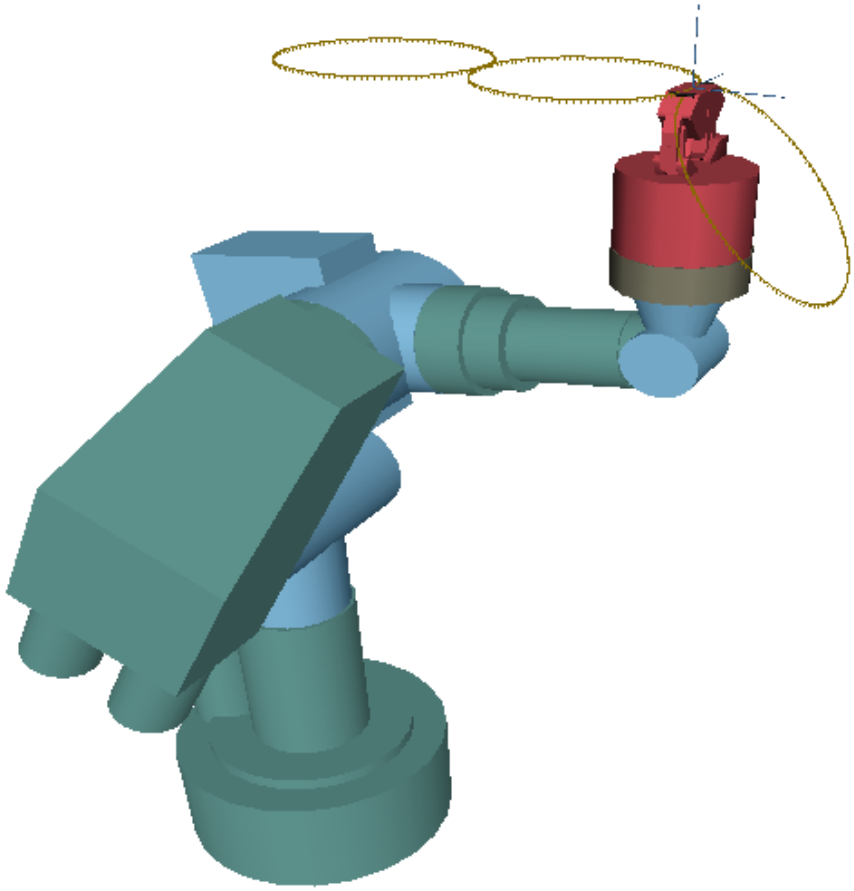


Abbildung C.4: Darstellung der Endeffektorpositionen und der Roboterstellung des Manutec r2 am Ende von Bahn 4.

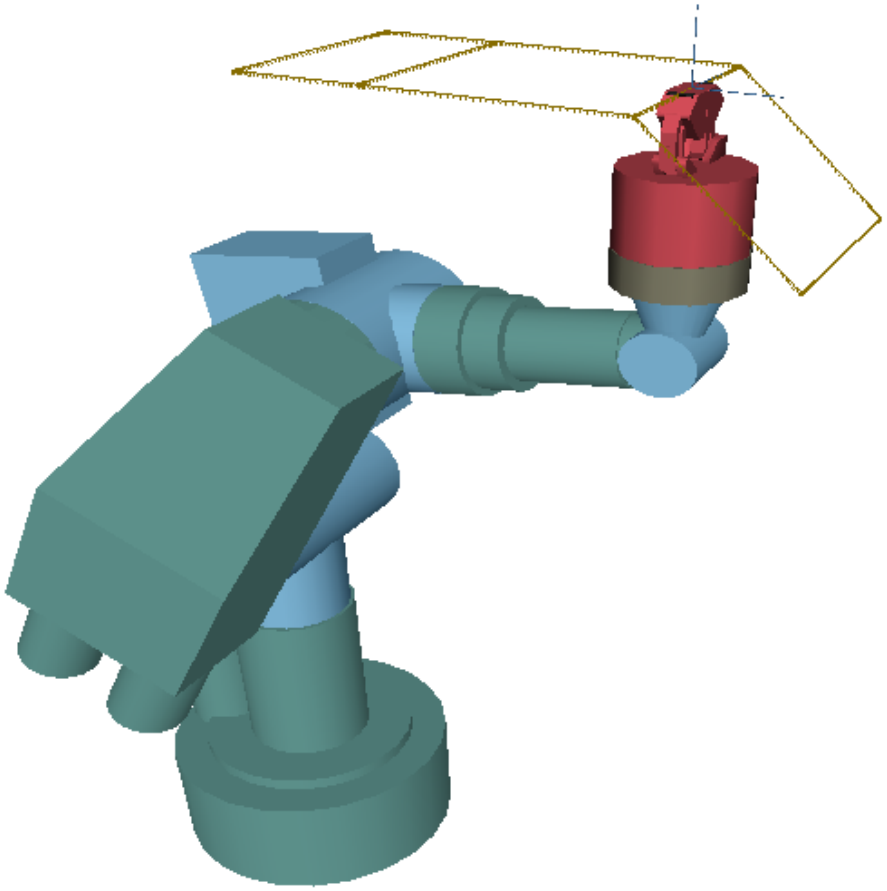


Abbildung C.5: Darstellung der Endeffektorpositionen und der Roboterstellung des Manutec r2 am Ende von Bahn 5.

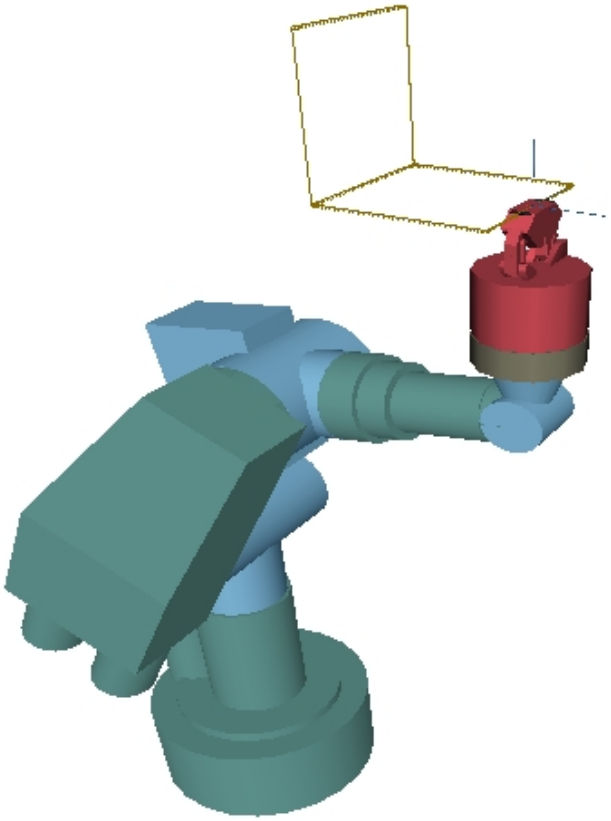


Abbildung C.6: Darstellung der Endeffektorpositionen und der Roboterstellung des Manutec r2 am Ende von Bahn 6.

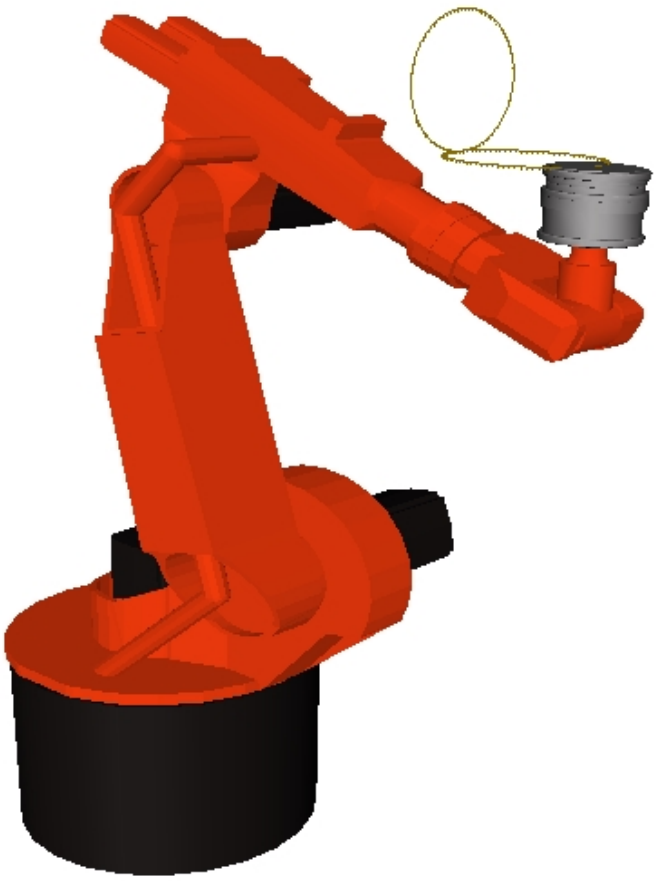


Abbildung C.7: Darstellung der Endeffektorpositionen und der Roboterstellung des Kuka KR6/1 am Ende von Bahn 3.

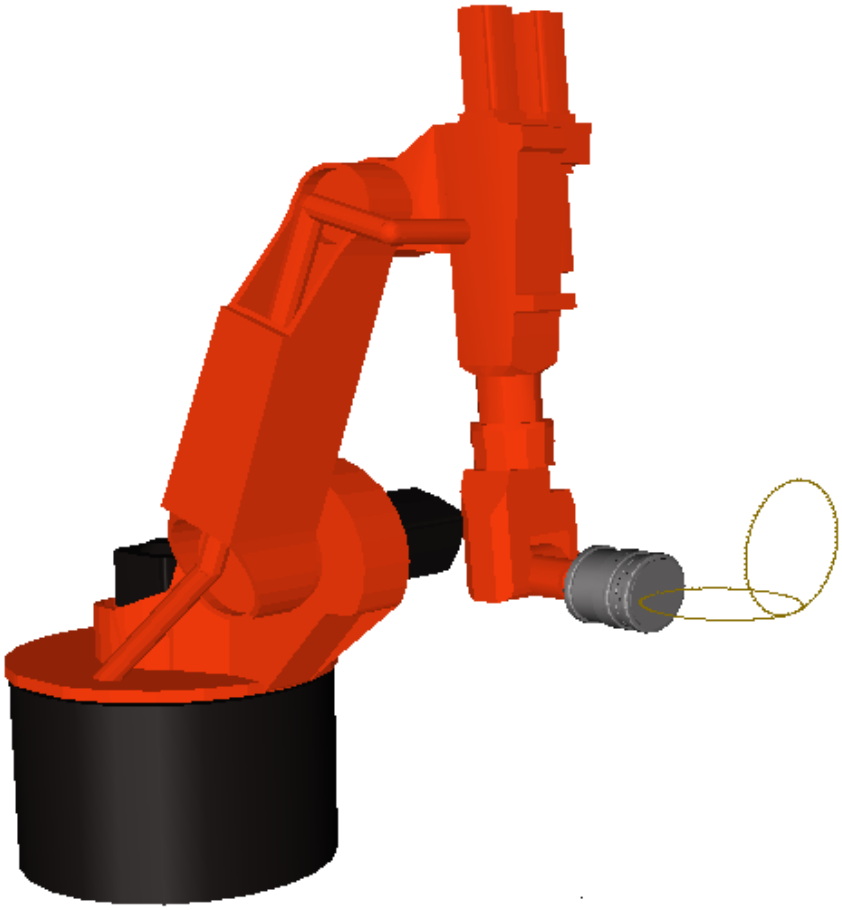


Abbildung C.8: Darstellung der Endeffektorpositionen und der Roboterstellung des Kuka KR6/1 am Ende von Bahn 3a.

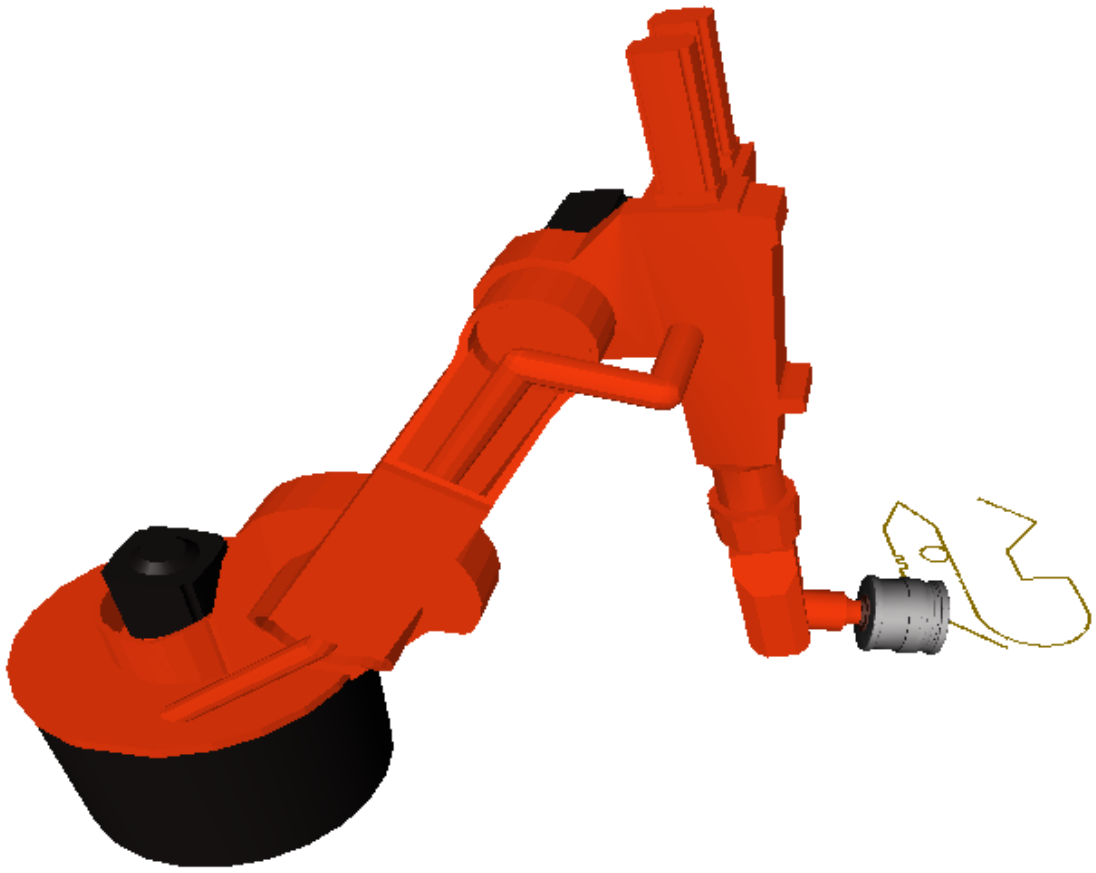


Abbildung C.9: Darstellung der Endeffektorpositionen und der Roboterstellung am Ende der Schmid'schen Bahn (Die Bahn enthält vor dem letzten Bahnsegment eine Richtungsumkehr)

Anhang D

Sensorbasierte Erzeugung von Bahnen

D.1 Anwendung des Kraftsensors zur Bahnplanung für die Konturverfolgung

Für die Konturverfolgung aufgrund der Daten eines Kraft- / Momentensensors muss zunächst eine Ebene[1] definiert werden, in der die Bewegung stattfinden soll. Für die Aufgabe nach Abb. 3.17 ist dies die x-y-Ebene des Roboters. In dieser Ebene wird ein Freiheitsgrad durch die Sensordaten vorgegeben, während die andere Komponente eine Positionsvorgabe enthält. In der zu dieser Ebene senkrechten Richtung soll keine Bewegung stattfinden. Das bedeutet, dass der durch manuelles Heranfahren erzeugte Anfangswert in dieser Komponente während der gesamten Trajektorie erhalten bleiben soll.

Die Aufteilung in die Komponenten erfolgt in jedem Schritt aufgrund des Kraftvektors $\mathbf{F}$, der den normierten Richtungsvektor $\mathbf{r}_n$ normal zur Oberfläche erzeugt (siehe Abb. D.1). Dabei ist angenommen, dass Kräfte nur in dieser Richtung auftreten können, es soll also weder Reibung betrachtet werden, noch soll der Stift in mehreren Freiheitsgraden eingeschränkt sein, wie im Falle einer zu stark konkaven Krümmung oder einer Innenkante.

Die Soll-Kraft F_d ist dementsprechend nur betragsmäßig vorgegeben und hat als Richtung die aktuelle Kraftrichtung. Damit wird die kraftgeregelte Komponente (Komponente normal zur Kontur) der kartesischen Soll-Position $\mathbf{x}_d$ zu

[1] Prinzipiell ist es zulässig, dass die Ebene sich während der Bewegung ändert. Dies wird jedoch nicht weiter verfolgt.

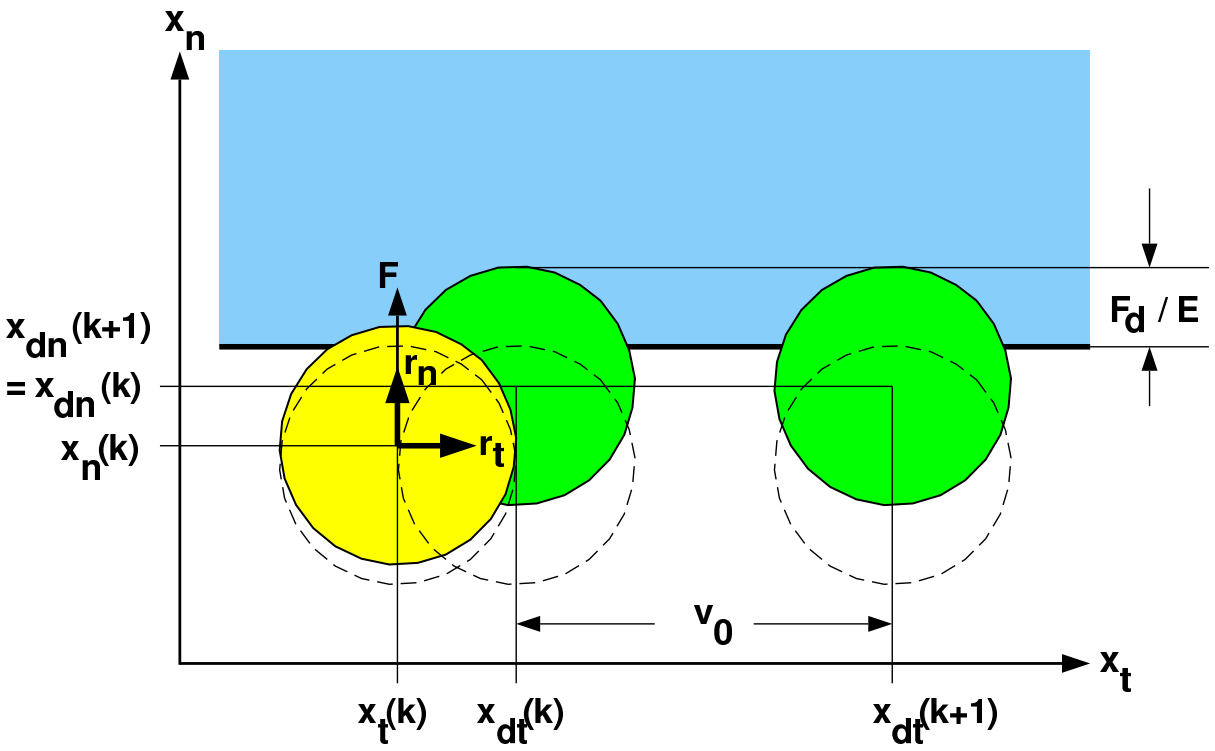


Abbildung D.1: Aufgrund der Robotergelenkwerte unter Vernachlässigung der Nachgiebigkeit transformierte Ist-Position $\mathbf{x}(k)$ (gelb) und Soll-Position $\mathbf{x}_d(k)$ bzw. $\mathbf{x}_d(k+1)$ (grün) eines runden Stiftes beim Ertasten einer Kante (gestrichelt = tatsächliche Ist- und Soll-Positionen des Stiftes)

$$x_{dn}(k+1) = \mathbf{r}_n^T(k+1) \cdot \mathbf{x}(k+1) + \frac{F_d(k+1) - |\mathbf{F}(k+1)|}{E}. \tag{D.1}$$

Dabei ist E die Steifigkeit, die kartesisch wirkt und in den beiden Richtungen der Bewegungsebene gleich ist. Gleichung D.1 bildet also die Differenz aus Soll- und Ist-Kraftbetrag und addiert das Positionsinkrement, das diesem Kraftfehler entspricht, auf die entsprechende Komponente der Ist-Position.

Die Soll-Position für den Zeitpunkt $k+1$ ist also erst definiert, wenn sowohl die aktuelle Position als auch die Sensorwerte für den gleichen Zeitpunkt vorliegen. Dabei ist darauf zu achten, dass zwischen beiden Messungen keine zeitliche Differenz aufgrund von Messtotzeiten liegt (Sensorvorverarbeitung siehe [150]).

Für die positionsgeregelte Komponente (Komponente tangential zur Kontur) ergibt sich

$$x_{dt}(k+1) = \mathbf{r}_t^T(k+1) \cdot \mathbf{x}_d(k) + v_0 \tag{D.2}$$

mit einem kartesisch konstanten Vorschub v_0 und einem normierten Richtungsvektor $\mathbf{r}_t$, der in der x-y-Ebene senkrecht zum Kraftvektor, also tangential zur Oberfläche steht (siehe Abb. D.1).

Aufgrund der Eigenschaften der Richtungsvektoren lässt sich jeder zweidimensionale Positionsvektor durch die beiden Komponenten ausdrücken, also

$$\mathbf{x}_d = \mathbf{r}_n^T \cdot \mathbf{x}_d \cdot \mathbf{r}_n + \mathbf{r}_t^T \cdot \mathbf{x}_d \cdot \mathbf{r}_t = x_{dn} \cdot \mathbf{r}_n + x_{dt} \cdot \mathbf{r}_t. \tag{D.3}$$

Damit liefern Gleichung (D.1) und (D.2) den Soll-Vektor:

$$\begin{aligned} \mathbf{x}_d(k+1) \;=\; & \mathbf{r}_n^T(k+1) \cdot \mathbf{x}(k+1) \cdot \mathbf{r}_n(k+1) \\ & + \frac{F_d(k+1) \cdot \mathbf{r}_n(k+1) - \mathbf{F}(k+1)}{E} \\ & + \mathbf{r}_t^T(k+1) \cdot \mathbf{x}_d(k) \cdot \mathbf{r}_t(k+1) + v_0 \cdot \mathbf{r}_t(k+1) \end{aligned} \tag{D.4}$$

Wenn man Gleichung (D.3) auch für $\mathbf{x}$ einsetzt, ergibt sich daraus:

$$\begin{aligned} \mathbf{x}_d(k+1) \;=\; & \mathbf{x}(k+1) + \frac{F_d(k+1) \cdot \mathbf{r}_n(k+1) - \mathbf{F}(k+1)}{E} \\ + \; & \mathbf{r}_t^T(k+1) \cdot (\mathbf{x}_d(k) - \mathbf{x}(k+1)) \cdot \mathbf{r}_t(k+1) + v_0 \cdot \mathbf{r}_t(k+1) \end{aligned} \tag{D.5}$$

Durch diese Gleichung lässt sich nachträglich die Soll-Bahn bestimmen und abspeichern.

Problematisch ist der Ansatz bei Ecken der Kontur. Aufgrund des endlichen Stabdurchmesser kommen aber keine Ecken vor. Stattdessen ist durch den Radius von 7.5 mm auch bei elastischem Ausweichen des Stabes ein minimaler Kurvenradius von etwa 3 mm (siehe Abb. D.2), bei $v_0 = 0.6$ mm/Schritt also etwa 5 Abtastschritten gegeben, was das Problem entschärft. Lediglich konkave Ecken führen zu zeitweise undefinierten Kraftrichtungen. Das wird hier aber nicht weiter behandelt.

Entsprechend Gleichung (D.5) lassen sich auch die zur Vorsteuerung benötigten künftigen Soll-Positionen vorhersagen:

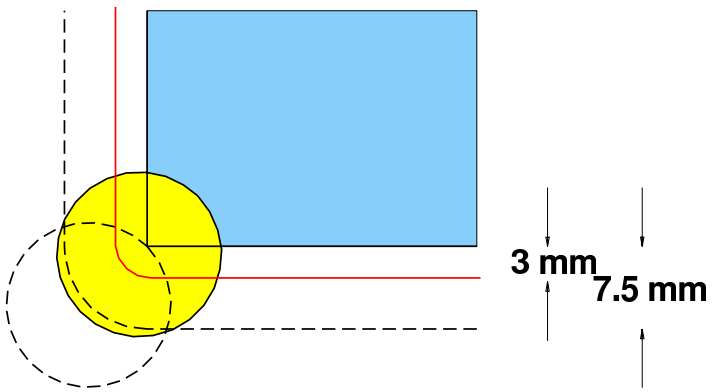


Abbildung D.2: Tatsächliche Bahn des Stabmittelpunktes (schwarz gestrichelt) und durch die Soll-Kraft und die Nachgiebigkeit definierte ideale Bahn des Roboters (rot durchgezogen) beim Ertasten einer Ecke mit einem runden Stift (gelber Kreis = scheinbare Position des Stabes)

$$\begin{aligned} \hat{\mathbf{x}}_d(k+i) &= \mathbf{x}(k+1) + \frac{F_d(k+1)\cdot\mathbf{r}_n(k+1) - \mathbf{F}(k+1)}{E} \\ &+ \mathbf{r}_t^T(k+1)\cdot(\mathbf{x}_d(k) - \mathbf{x}(k+1))\cdot\mathbf{r}_t(k+1) + i\cdot v_0\cdot\mathbf{r}_t(k+1) \end{aligned} \tag{D.6}$$

Um nicht für jedes i eine Rückwärtstransformation zu durchlaufen, werden im Schritt k nur $\hat{\mathbf{q}}_d(k+10)$ und $\mathbf{q}_d(k)$ berechnet. Die anderen Gelenkwerte $\hat{\mathbf{q}}_d(k+i)$ werden dann durch lineare Interpolation zwischen den beiden Positionen erzeugt. Die dadurch entstehenden Abweichungen sind gering.

Ein anderer Ansatz, bei dem entgegen Gleichung (D.1) auch die Krümmung der Oberfläche berücksichtigt wird, wurde getestet, erweist sich aber bei Vorhersagen über viele Schritte als zu störungsempfindlich, da die Krümmung aus der Differenz von Richtungsvektoren berechnet wird, die ihrerseits nicht genau genug bestimmt werden können.

Das Problem liegt in der Vorhersage des Konturverlaufs, was grundsätzlich nicht fehlerfrei möglich ist und daher während des Ertastens Regelfehler verursacht. Beim Wiederholen der ertasteten Bahn (Gleichung (D.5)) kann dagegen die Genauigkeit offline programmierter Bahnen erreicht werden.

D.1.1 Stabilität der Kraftregelung

Bei der hier verwendeteten hierarchischen Struktur aus Abschnitt 4.2 handelt es sich durch die Rückführung der Sensordaten um eine Regelung. Die oben getroffene Aussage, dass die Stbiliät gewährleistet ist, sofern sowohl die Realisierung des *idealen Roboters* als auch die Bahnplanung jeweils optimal ausgelegt werden, lässt sich nun präzisieren.

Bezüglich der Bahnplanung ist die Stabilität gewährleistet, wenn

$$\hat{E} > 0.5 \cdot E, \tag{D.7}$$

wenn also die angenommene Steifigkeit des Sensors mehr als halb so groß wie die tatsächliche Steifigkeit ist. Zur Erläuterung wird die Vorhersagegleichung

$$\hat{\mathbf{x}}_d = \mathbf{x} + (\mathbf{F}_d - \mathbf{F}(\mathbf{x}))/\hat{E} + \cdots \cdot \mathbf{r}_t \tag{D.8}$$

für die Position $\hat{\mathbf{x}}_d$ betrachtet, bei der der Kraftsollvektor $\mathbf{F}_d$ erreicht wird. Dies wird in die Kraftgleichung

$$\mathbf{F}(\hat{\mathbf{x}}_d) = \mathbf{F}(\mathbf{x}) + E \cdot \mathbf{r}_n^T \cdot (\hat{\mathbf{x}}_d - \mathbf{x}) \cdot \mathbf{r}_n \tag{D.9}$$

eingesetzt:

$$\mathbf{F}(\hat{\mathbf{x}}_d) = \mathbf{F}(\mathbf{x}) + (\mathbf{F}_d - \mathbf{F}(\mathbf{x})) \cdot E/\hat{E} \tag{D.10}$$

Bei $E/\hat{E} > 2$ ist der Koeffizient von $\mathbf{F}(\mathbf{x})$ betragsmäßig größer als 1, was zu einem instabilen System führt. Dagegen bringt eine zu groß angenommene Steifigkeit keine Stabilitätsprobleme.

D.2 Verwendung eines Bildverarbeitungssystems als prädiktiven Sensor

Für die Repräsentation einer Linie im Bild werden zunächst einzelne Punklte der Linie detektiert. Aus diesem Punkten wird ein Polynom gebildet. Dieses Polynom ist dann die Basis zur Berechnung der Soll-Positionen $\mathbf{x}_d(k+i)$ der nächsten n_w Abtastschritte.

Die Kamera wird so ausgerichtet, dass die Linie etwa senkrecht verläuft. Nun werden einige Bildzeilen ausgewählt, an denen die Position der Linie festgestellt werden soll. Die erste Zeile entspricht dem TCP, also $\mathbf{x}$. Die übrigen Zeilen decken den Prädiktionsbereich ab, also den Bereich bis ungefähr $\mathbf{x}_d(k+n_w)$, sofern dies aufgrund des begrenzten Öffnungswinkels der Kamera möglich ist. Diese Zeilen variieren in Abhängigkeit der Geschwindigkeit und der Bewegungsrichtung (vorwärts oder rückwärts).

Für diese Bildzeilen wird ein Kontrastbild berechnet, wie es zur Demonstration für den gesamten Bildbereich in Abb. 6.17 dargestellt ist. Dieses Kontrastbild entsteht z. B. durch Filterung des Bildes mit einem Sobel Operator

$$\begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix} \tag{D.11}$$

Im sich ergebenden Bild werden Hell-Dunkel-Übergänge durch betragsmäßig große Werte repräsentiert. Durch Suche der Maximalwerte oder der dazugehörigen Flanken werden die Spaltenwerte der zu den interessierenden Zeilen gehörigen Linienpunkte ermittelt. Dadurch ergibt sich ein Punkt der gesuchten Linie im Bild.

Bei Kenntnis des ungefähren Abstands der Linie von der Kamera kann daraus die Position des Linienpunktes im Raum ermittelt werden. Aus den verschiedenen Linienpunkten wird dann im Raum ein Polynom gebildet. Die als Gerade programmierte Referenzbahn gibt zunächst alle Komponenten der Soll-Bahn vor. Die Komponente senkrecht zur Linie wird dann durch Auswertung des Polynoms für den aktuellen und die zukünftigen Abtastpunkte der Referenzbahn überschrieben.

Dabei geschieht die Auswertung des Polynoms in jedem Abtastschritt, während die Bestimmung der Linienrepräsentation asynchron im 50 Hz-Takt der Bildverarbeitung erfolgt. Dieser Takt ist möglich, da im Field-Modus gearbeitet wird und die Berechnung wegen der Beschränkung auf wenige Bildzeilen sehr schnell geht.

Zur Kalibrierung der Kamera werden zunächst die internen Parameter bestimmt, insbesondere die Linsenverzeichnung [134]. Kurz vor dem eigentlich Experiment wird die Entfernung der Linie durch eine Skalierungsbewegung analog zu [11] ermittelt. Dabei wird die Kamera senkrecht zur erwarteten Linienrichtung verschoben. Die Orientierung der Kamera wird während der gesamten Anwendungsbewegung überprüft (ausführlicher in [156]), da sie sich ggf. aufgrund von Nachgiebigkeiten oder einer ungenauen kinematischen Transformation des Roboters ändern kann. Die Orientierung muss genau bekannt sein, da auch Punkte am Bildrand mit Sub-Pixel-Genauigkeit vermessen werden sollen.

Eine Filterung der Bildinformation erfolgt nur in Bezug auf die räumliche Lage der Linie im Raum. Direkte Filterung der gefundenen Punkte im Bild würde dagegen die Dynamik beschränken.

Durch die zu Beginn durchgeführte Skalierung ist die Stabilität theoretisch gewährleistet, da tatsächlicher und ermittelter Abstand der Linie sich nicht um mehr als den Faktor 2 unterscheiden (vergl. Abschnitt D.1.1). Durch die Sensorrückführung können allerdings Schwingungen angeregt werden, die ohne einen Beobachter nach Abb. 4.17 zu einer Verschlechterung des Verhaltens führen können. Im Extremfall, bei sehr stark gekrümmter Linie, können die zu vermessenden Punkte den im Bild ausgewerteten Bereich verlassen. Eine Begrenzung der Bandbreite der Regelung wird trotzdem nicht vorgenommen, da sie die hohe Regelgüte einschränken würde. Eine Begrenzung der Sensorkorrektur auf z. B. ± 200 mm ist aber möglich.

Lebenslauf

Name	Friedrich Konrad Hermann Lange	
Wohnort	Germering	
Familienstand	verheiratet	
Geburt	04.06.1955	in Kassel
Schulabschluss	1974	Abitur
Wehrdienst	1974 - 1976	
Studium	1976 - 1982	Fachrichtung Elektrotechnik, Studienrichtung Regelungstechnik an der Technischen Hochschule Darmstadt (Abschluss: Dipl.-Ing.)
Forschungstätigkeit	seit 1983	als wissenschaftlicher Mitarbeiter des DLR Oberpfaffenhofen, Institut für Robotik und Mechatronik

Arbeitsgebiete

- Assoziativspeicher, Neuronale Netze, Lernen von nichtlinearen Abbildungen,
- adaptive und lernende Regelungssysteme, Verbesserung der Bahngenauigkeit von Robotern, Lernen von sensorgestützten Roboterbewegungen (z. B. Konturverfolgung),
- Architekturen für Roboterregelungen und Sensordatenverarbeitung,
- sensorgestützte Erfassung und Korrektur von Roboterbewegungen (z. B. durch Bildverarbeitung)