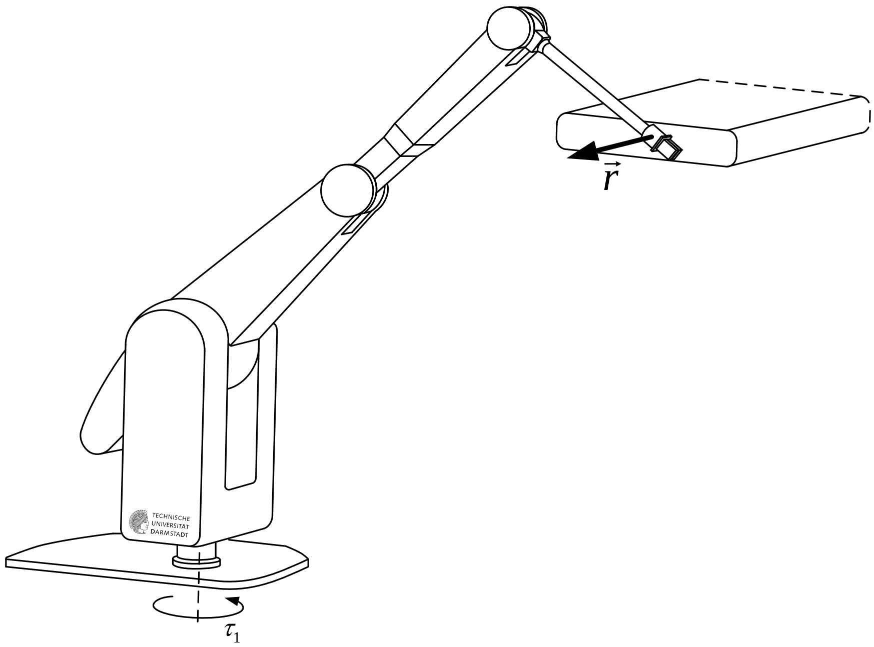
Technische Universität Darmstadt
Fachbereich Informatik
Fachgebiet Simulation, Systemoptimierung und Robotik
Prof. Dr. Oskar von Stryk

Kollisionserkennung am gelenkelastischen Manipulator BioRob mittels Residuen basierend auf dem verallgemeinerten Impuls



Bachelor Thesis

Jérôme Kirchhoff

Oktober 2008

Betreuer: Dipl.-Ing. Thomas Lens

Ehrenwörtliche Erklärung

Hiermit versichere ich, dass die vorliegende Bachelor Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt wurde. Alle Stellen, die aus den Quellen entnommen wurden, sind als solche kenntlich gemacht worden. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, Oktober 2008

Jérôme Kirchhoff

Zusammenfassung

Um die höchst mögliche Sicherheit bei der Mensch-Roboter-Interaktion zu gewährleisten, muss sichergestellt sein, dass ungewünschte Kollisionen erkannt werden. Ist eine Kollision erkannt, muss auf diese angemessen reagiert werden. Schon relativ leichte Roboter können bei Kollisionen Frakturen beim Menschen verursachen. Diese Arbeit stellt ein Verfahren basierend auf dem verallgemeinerten Impuls vor, mit dem es möglich ist Kollisionen nicht nur zu erkennen, sondern auch zu bestimmen welches Glied an der Kollision beteiligt ist. Zudem wird bewertet wie der Leichtbaumanipulator BioRob auf Kollisionen reagiert, und welche Gefahr überhaupt von ihm ausgeht. Ein Vorteil vom BioRob könnte seine Elastizität darstellen, welche die auftretende Kraft bei einer Kollision verzögert abgibt. Aber auch verschiedene Reaktionsstrategien zum vermindern der Kollisionsschäden werden vorgestellt und diskutiert. Um das Verfahren zu beurteilen werden der Manipulator sowie die Kollisionsdetektionsstrategien in Matlab/Simulink modelliert. Das ermöglicht eine erste Bewertung über die Effektivität des Verfahrens.

Schüsselwörter: Kollisionserkennung, Verallgemeinerter Impuls, Fault Detection and Isolation, Gelenkelastizität, Leichtbaumanipulator

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungen und Formelzeichen

$\boldsymbol{B}$	[kg m^2]	Positiv definite Diagonalmatrix der Motorträgheiten
$\boldsymbol{C}(q,\dot{q})$	[Nm]	Matrix der auf den Manipulator wirkenden Corioliskräfte
$\boldsymbol{d}$	[Nm s/rad]	Dämpfung/Reibung im Gelenk
$\boldsymbol{D}$	[Nms/rad]	Diagonalmatrix der viskosen Gelenkreibung
$\boldsymbol{d_e}$	[Nms/rad]	Vektor der Dämpfungskonstanten im Feder-Dämpfer-Modell
δX	[m]	Virtuelle Verrückung im Arbeitsraum
δq	[m]	Virtuelle Verrückung im Gelenkraum
$\boldsymbol{e_i}$		Einheitsvektor $e_i = \underset{1 \dots i \dots n}{[0 \dots 1 \dots 0]^T}$
$\boldsymbol{F}(q,\dot{q})$	[Nm s/rad]	Diagonalmatrix der Reibungs- und Dämpfungskräfte
$\boldsymbol{F_k}$	[N]	Vektor der bei einer Kollision von außen wirkenden Kraft
$\boldsymbol{g}$	[m/s^2]	Erdschwerebeschleunigung
$\boldsymbol{G}(q)$	[Nm]	Matrix der auf den Manipulator wirkenden Gravitationskräfte
$G_1(s)$		Übertragungsfunktion des Übertragungsgliedes (Filter)
$\boldsymbol{i}$		Vektor der Übersetzungsverhältnisse in den Getrieben
$\boldsymbol{I}$		4x4 Einheitsmatrix
$\boldsymbol{J}(q)$		Jakobimatrix des Manipulators
$\boldsymbol{J}$	[kg m^2]	Trägheitsmomente der Glieder um die Gelenke
$\boldsymbol{J_m}$	[kg m^2]	Vektor der Trägheitsmomente der Motoren um ihre Rotationsachse
$\boldsymbol{J_g}$	[kg m^2]	Vektor der Trägheitsmomente der Getriebe um ihre Rotationsachse bezüglich der Abtriebsseite
$\boldsymbol{K}$	[Nm/rad]	positiv definite Diagonalmatrix der Gelenksteifigkeiten (Federkonstanten)
$\boldsymbol{k_e}$	[Nm/rad]	Vektor der Federkonstanten im das Feder-Dämpfer-Modell
$\boldsymbol{K_I}$		Diagonalmatrix der Verstärkungsfaktoren des Residuums
$\boldsymbol{K_R}$		Diagonalmatrix der Verstärkungsfaktoren der Kollisionsstrategien 3 und 4
$\boldsymbol{L}$	[Nm]	Drehimpuls eines starren Körpers (z.B. Motorwelle)
$\boldsymbol{l}$	[m]	Vektor der Längen der Glieder

$\boldsymbol{m}$	[kg]	Vektor der Massen der Glieder
$\boldsymbol{p}$		Verallgemeinerte Impuls
$\boldsymbol{q}$	[rad]	Vektor der Winkelpositionen der Gelenke
$\dot{\boldsymbol{q}}$	[Nm/s]	Vektor der Winkelgeschwindigkeiten der Gelenke
$\ddot{\boldsymbol{q}}$	[Nm/s²]	Vektor der Winkelbeschleunigungen der Gelenke
$R_1(s)$		Laplacetransformierte Ausgangsgröße
$\boldsymbol{r}$	[Nm]	Residuumsvektor
$\boldsymbol{r_c}$	[m]	Vektor der Abstände zwischen Gelenk $i+1$ und Schwerpunkt des i-ten Gliedes
$\boldsymbol{T}$	[s]	Diagonalmatrix der Zeitkonstanten des Residuums
${}^0\boldsymbol{T}_i(\boldsymbol{q})$		homogene (4x4)-Transformationsmatritzen
$\boldsymbol{\tau}$	[Nm]	Vektor der Drehmomente der Gelenke
$\hat{\boldsymbol{\tau}}$	[Nm]	Vektor der Drehmomente der Gelenke, welche über das Roboterdynamikmodell ermittelt wurden
$\boldsymbol{\tau_d}$	[Nm]	Vektor der Kräfte, welche in den Dämpfern der Feder-Dämpfer-Modelle der elastischen Gelenkkopplungen mit den Motoren wirken
$\boldsymbol{\tau_{ext}}$	[Nm]	Vektor der Drehmomente in den Gelenken, verursacht von einer externen Kraft
$\boldsymbol{\tau_{el}}$	[Nm]	Vektor der Kräfte, welche in den Federn der Feder-Dämpfer-Modellen der elastische Gelenkkopplungen mit den Motoren wirken
$\boldsymbol{\tau_f}$	[Nm]	Vektor der Gesamtkraft, welche in den Feder-Dämpfer-Modellen der elastischen Gelenkkopplungen mit den Motoren wirken
$\boldsymbol{\tau_k}$	[Nm]	Vektor der Drehmomente in den Gelenken, verursacht durch eine Kollision
$\boldsymbol{\tau_m}$	[Nm]	Vektor der Drehmomente der Motoren
$\boldsymbol{\theta}$	[rad]	Vektor der Winkelpositionen der Motoren
$\dot{\boldsymbol{\theta}}$	[Nm/s]	Vektor der Winkelgeschwindigkeiten der Motoren
$\ddot{\boldsymbol{\theta}}$	[Nm/s²]	Vektor der Winkelbeschleunigungen der Motoren
$U_1(s)$		Laplacetransformierte Eingangsgröße

1. Einleitung und Überblick

In modernen Fabrikhallen, in denen Industrieroboter die Arbeit des Menschen übernommen haben, kann man beobachten, wie wichtig das Thema Sicherheit für den Einsatz solcher Roboter ist. Hier wird penibel darauf geachtet, dass sich die Arbeitsbereiche von Mensch und Maschine nicht überschneiden. Um dies zu gewährleisten, werden die Arbeitsbereiche der Roboter durch einen Gitterkäfig eingezäunt oder durch Lichtschranken geschützt. Wird die Tür zum Käfig geöffnet oder eine der Lichtschranken unterbrochen, stellt der Roboter sofort die Arbeit ein, um möglichen Verletzungen vorzubeugen.

An diesem Beispiel ist zu erkennen, dass für einen gewissen Sicherheitsstandard nicht nur in den Roboter selbst, sondern auch in weitere externe Sensoren zu investieren ist. Dies scheint bei fest installierten Industrierobotern auch sinnvoll, da sich die Kollisionskräfte schon bei „kleinen“[1] Robotern in einem Bereich bewegen, der zu schweren Verletzungen führen kann.

Betrachtet man jedoch den vom Fachgebiet Simulation, Systemoptimierung und Robotik entwickelten Leichtbauarm BioRob, so sind die oben beschriebenen Maßnahmen unpraktikabel. Der Manipulator orientiert sich an der Reichweite und Traglast eines menschlichen Arms, ebenso flexibel soll dieser auch einsetzbar sein. Das bedeutet, der Roboterarm ist leicht zu transportieren und führt Arbeiten schon nach einer kurzen Einrichtungszeit aus. Das Aufbauen aufwändiger Sensorik zur Vermeidung unerwünschter Kollisionen spricht gegen die Portabilität und die kurzen Einrichtungszeiten vom BioRob. Aus sicherheitstechnischen Gesichtspunkten sind beim BioRob solch aufwändige Maßnahmen nicht unbedingt nötig, da dieser durch seine Leichtbauweise (Eigengewicht von 3.5 kg) und Nachgiebigkeit passiv sicher ist. Auch darf die Flexibilität nicht durch die Bedingung eingeschränkt werden, dass sich in der Nähe des Roboters keine Personen aufhalten dürfen.

Um diesem Problem zu entgehen, muss also eine andere Lösung gefunden werden, die Kollisionen im gemeinsamen Arbeitsbereich von Mensch und Roboter erkennt. Nach dem Auftreten der Kollision besteht die wichtigste Aufgabe darin, die Kollision zu entdecken. Hierbei muss berücksichtigt werden, dass diese an jedem Punkt des Armes aufgetreten sein kann. Ein dann intelligentes Reagieren auf eine solche Situation ist zudem wichtig, ja sogar wesentlich für eine effiziente Mensch-Roboter-Interaktion. Dies könnte zum Beispiel durch Stoppen des Manipulators oder Herausfahren aus der Kollision geschehen. In dieser Ausarbeitung wird die Kollisionserkennung, wie in [3] für den Light-Weight-Robot III vom DLR beschrieben, mittels verallgemeinertem Impuls (generalized momentum) umgesetzt. Mit Hilfe dieses Ansatzes ist es nicht notwendig die Geometrie des Arbeitsbereichs vom BioRob zu ermitteln, um diesen dann

[1] Schon der Industrieroboter KUKA KR3-SI (Eigengewicht: 54kg, Traglast: 3kg) kann bei einer Endeffektorgeschwindigkeit von schon 1m/s Frakturen im Gesicht verursachen

mit rechenaufwendiger Bahnplanung in Bereiche aufzuteilen, die der Roboter während der Ausführung nicht ansteuern darf.

In Kapitel 2 wird die Problematik der Kollisionserkennung erläutert. Hierzu zählen auch die Schäden, die eine Kollision beim Menschen verursachen kann. Die Stärke der Schäden richtet sich dabei nicht nur nach der Geschwindigkeit des Manipulators, sondern auch nach der Art des Aufschlags (Stumpf, Schnitt, usw.), sowie der Position des Aufschlags (Brust, Hals, Kopf, usw.). Nachdem die Problemstellung dargelegt wurde, folgt eine Übersicht über aktuelle Lösungsansätze.

Um den gewählten Lösungsansatz auf dem BioRob anzuwenden, wird in Kapitel 3 kurz dessen Aufbau und Modellierung dargestellt. Anschließend folgt die Herleitung der Formeln anhand des BioRob Modells, basierend auf dem verallgemeinerten Impuls. Mit Hilfe dieser Formeln ist es dann möglich, das Auftreten einer Kollision entlang des Manipulators zu erkennen. Im nächsten Schritt wird darauf aufbauend erläutert, welche Reaktionsstrategien möglich sind, und wie das erlangte Wissen über die Kollision hierbei helfen kann.

Schließlich folgt in Kapitel 4 die Beschreibung der Simulation mit Matlab/Simulink. Mit Hilfe der Simulation ist es möglich das Verfahren soweit zu verfeinern, dass im Anschluss eine schnelle Implementierung auf dem Prototypen des BioRobs möglich ist. Dabei wird ein möglichst genaues Modell des Manipulators nachgebildet, um dort das Erkennen von Kollisionen zu testen. Auch die wirkenden Kräfte, aber vor allem die Wirkung der Reaktionsstrategien werden betrachtet. Nachdem die ausschlaggebenden Simulationen durchgeführt wurden, ist es möglich, über deren Auswertung die betrachteten Sicherheitsaspekte in Bezug auf den BioRob zu bewerten. Ferner kann auch beleuchtet werden, welche Reaktionsstrategie die effektivste ist und die es sich deshalb lohnt, auf dem Prototypen zu installieren.

Abschließend sind in Kapitel 5 alle wichtigen Punkte und Eigenschaften des Verfahrens noch einmal zusammengefasst. Als Ausblick wird zudem noch skizziert, welche Aufgaben nun mit Hilfe des vorgestellten Verfahrens erledigt werden können, damit der BioRob ein sicherer Arbeitskollege werden kann.

2. Problemstellung und aktuelle Ansätze

Wie in der Einleitung schon beschrieben, hat der Schutz des Menschen, mit dem sich ein Roboter den Arbeitsbereich teilt, absoluten Vorrang. In Anwendungen mit Industrierobotern sieht der Schutz meist so aus, dass der Arbeitsbereich des Menschen von dem des Roboters durch Gitter und Lichtschranken getrennt ist. Einlass zum Roboter erhält der Mensch durch eine Sicherheitstür. Wird diese geöffnet oder eine der Lichtschranken durchbrochen, erfolgt das sofortige Abschalten des Manipulators [8]. Doch die aktuelle Entwicklung zeigt, dass diese Trennung des Arbeitsbereichs häufig nicht mehr sinnvoll, oder gar nicht mehr aufrecht zu erhalten ist. Ein Beispiel dafür ist die ständige Weiterentwicklung von Servicerobotern, wie z.B. der im Sommer 2008 vorgestellte „Care-O-Bot 3“ des Fraunhofer-Institut für Produktionstechnik und Automatisierung (IPA) in Stuttgart. Damit ein Serviceroboter seine Aufgaben verrichten kann, ist es unbedingt notwendig, dass dies ohne Sicherheitsbedenken für die Personen in seiner Umgebung möglich ist. Im Falle des BioRobs gilt zudem, dass sein Einsatzgebiet[1] nicht durch aufwändige Aufbauten eingeschränkt werden soll. Daher kommen die oben genannten Sicherheitsvorkehrungen im Falle des BioRobs nicht in Frage.

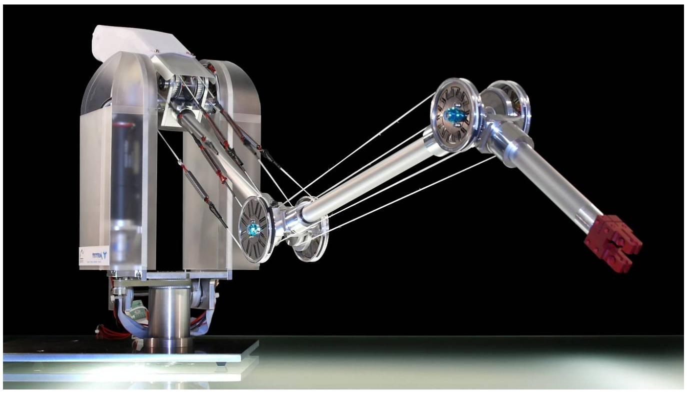


Bild 2.1.: Der gelenkelastische Roboterarm BioRob (c) Tetra GmbH

Man steht demnach vor der Aufgabe, mit sehr einfachen Mitteln hohe Sicherheit zu gewährleisten. Aber wann ist ein Roboter sicher? Um diese Frage zu beantworten, muss man zunächst betrachten, welchen Schaden oder welche Verletzung ein Manipulator überhaupt verursachen kann. Hierzu möchte ich kurz die sehr interessanten Ergebnisse von [5], [6], [7] und [8] darlegen.

[1] Das geplante Einsatzgebiet liegt z.B. am Fließband oder generell in Greifanwendungen in Zusammenarbeit mit anderen Personen, wie es auch bei Servicerobotern der Fall ist.

In [8] wurden folgende Erkenntnisse gewonnen: schnelle, starre Einschläge (bei ca. 2 m/s) rufen sehr hohe Kraftspitzen, in Abhängigkeit von der Einschlagsgeschwindigkeit, hervor. Die hierbei übertragene Kraft wird von dem Glied aber auch von den involvierten Motoren (inkl. Getrieben) verursacht. Zur Verdeutlichung ist in Bild 2.2 das Modell eines Motors, inkl. Getriebe und anschließendem Glied zu sehen, wobei $\boldsymbol{\tau_m}$ das Motormoment, $\boldsymbol{\theta}$, $\dot{\boldsymbol{\theta}}$, $\ddot{\boldsymbol{\theta}}$ die Motorposition, -geschwindigkeit, -beschleunigung , i die Getriebeübersetzung, $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, $\ddot{\boldsymbol{q}}$ die Gelenkposition, -geschwindigkeit, -beschleunigung, $\boldsymbol{J_m}$ die Motorträgheit, $\boldsymbol{J_g}$ die Getriebeträgheit und $\boldsymbol{J}$ die Trägheit des Gliedes bezüglich der Rotationsachse ist.

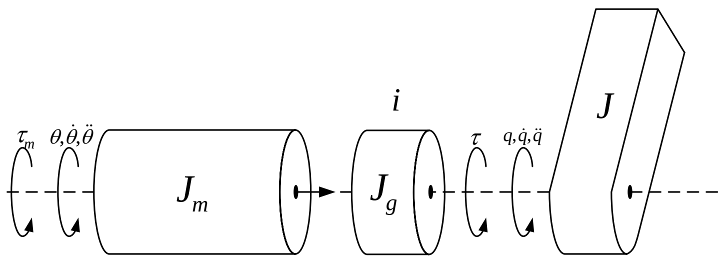


Bild 2.2.: Gelenkschema (Motor, Getriebe, Glied)

Der Zusammenhang zwischen Motorgeschwindigkeit und Gelenkgeschwindigkeit ist über die Getriebeübersetzung gegeben.

$$\dot{q} = i \cdot \dot{\theta} \tag{2.1}$$

Leitet man 2.1 einmal nach der Zeit ab, so erhält man

$$\ddot{q} = i \cdot \ddot{\theta} \tag{2.2}$$

Für die nächsten Überlegungen sei keine Last am Motor angeschlossen ($J = 0$) und die Getriebeträgheit vernachlässigbar ($J_g = 0$). Betrachtet man das Getriebe idealisiert mit einem Wirkungsgrad von 100 Prozent[2], so muss die Leistung über das Getriebe hinweg erhalten bleiben. Die Leistung des Antriebs wird durch $\dot{\theta} \cdot \tau_m$ und die des Abtriebs durch $\dot{q} \cdot \tau$ beschrieben [9]. Nach der eben genannte Forderung müssen diese beiden Terme gleich groß sein. Durch Gleichsetzen erhält man

$$\dot{q} \cdot \tau = \dot{\theta} \cdot \tau_m. \tag{2.3}$$

Gleichung 2.1 in 2.3 eingesetzt und umgestellt, führt zur Abhängigkeit der Momente vor und nach dem Getriebe

$$\begin{aligned} \dot{q} \cdot \tau &= \frac{\dot{q}}{i} \cdot \tau_m \\ \Rightarrow \tau_m &= \tau \cdot i \end{aligned} \tag{2.4}$$

[2]Der Fehler, welcher durch die Idealisierung entsteht, wird bei uns in die Reibung eingerechnet

Um die wirkende Trägheit des Motors auf das abtriebsseitige Drehmoment τ zu ermitteln, kann man 2.2 und 2.4 in die Motorgleichung $J_m\ddot{\theta} = \tau_m$ [9] einsetzen.

$$\begin{aligned} J_m\ddot{\theta} &= \tau_m \\ J_m\frac{\ddot{q}}{i} &= i\tau \\ J_m\ddot{q} &= i^2\tau \\ \tau &= \frac{J_m}{i^2}\ddot{q} \end{aligned} \tag{2.5}$$

In Gleichung 2.5 ist zu sehen, dass das antriebsseitige Trägheitsmoment des Motors ebenfalls über das Getriebe hinweg verändert wird. Die echt wirkende abtriebsseitige Motorträgheit $\frac{J_m}{i^2}$ bezeichnet man auch als „reflektierte Trägheit“ (reflected inertia). Diese Ungleichheit zwischen an- und abtriebsseitig wirkender Motorträgheit ist nicht zu vernachlässigen, denn durch die Leichtbauweise und hohe Übersetzung ins Langsame ist das Trägheitsmoment des Gliedes J sogar kleiner als $\frac{J_m}{i^2}$, obwohl J_m sehr klein ist.

Da wir nun das abtriebsseitig wirkende Moment des Motors kennen, können wir auch die restlichen Trägheiten betrachten. Für $J_g \neq 0$ und $J \neq 0$ können diese wirkenden Trägheiten in τ einfach addiert werden. Somit erhalten wir das endgültig wirkende Moment τ inkl. aller Trägheitsmomente:

$$\tau = (\frac{J_m}{i^2} + J_g + J)\ddot{q}. \tag{2.6}$$

Wir können in diesem Fall die Trägheitsmomente addieren, weil sie alle abtriebsseitig bzgl. derselben Achse gebildet werden. Man kann sich dazu veranschaulichen, dass die Volumenintegrale zur Ermittlung der einzelnen Trägheitsmomente entlang der Rotationsachse zu einem Integral zusammengefasst werden können. In diesem Integral müssen dann die jeweiligen verschiedenen Volumendichten berücksichtigt, und die so ermittelten einzelnen Integrale miteinander addiert werden.

Da wir nun die an einer Kollision beteiligten Kräfte kennen, können wir eine solche genauer betrachten. Bei starren Manipulatoren wird die gesamte im Gelenk wirkende Kraft auf einmal während der Kollision übertragen (also wirken die Trägheitsmomente $\frac{J_m}{i^2} + J_g + J$). Aber schon bei geringen Gelenkelastizitäten dreht der Motor nach einer Kollision ein Stück weiter. Dieses Weiterdrehen wird von der Elastizität bestimmt, denn je elastischer die Kopplung von Gelenk und Motor ist, desto weiter kann er drehen, bevor die Arretierung des Gliedes sich auf den Motor auswirkt. Solch ein Verhalten hat zur Folge, dass das Trägheitsmoment vom Motor/Getriebe ($\frac{J_m}{i^2} + J_g$) erst verzögert zum Trägheitsmoment des Glieds (J) in die Kollision abgegeben wird.

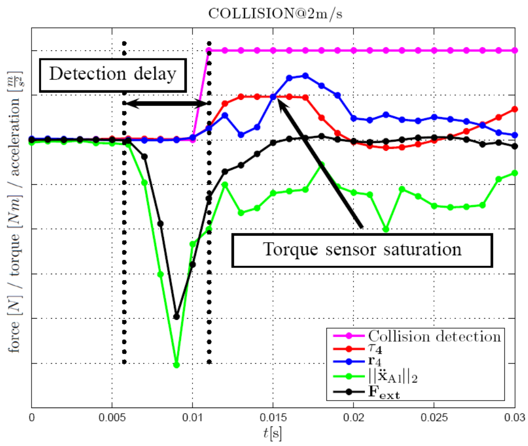


Bild 2.3.: Kraftübertragung während einer Kollision bei $2\frac{m}{s}$ (aus [8])

Genau dieser Zusammenhang ist in Bild 2.3 zu sehen. Die schwarze Linie zeigt die extern aufgezeichnete Kraft. Diese Kraftspitze entlädt sich hier von 4 bis 10ms. Die rote Linie beschreibt das im Gelenk wirkende Moment (dabei weist hier der Momentensensor eine Sättigung auf). Die Kollision bremst das Glied stark ab und verursacht damit unmittelbar eine Spitze im auf das Glied einwirkenden Moment (schwarze Linie). Eine Erhöhung des im Gelenk wirkenden Moments (rote Linie) erfolgt allerdings aufgrund der elastischen Kopplung zwischen Glied und Motor verzögert, da der Motor während der Kollision durch die Elastizität weitestgehend ungehindert weiter drehen kann. Die Stärke des vom Glied hervorgerufenen Moments wird von dessen Trägheitsmoment J bestimmt. Die Übertragung geschieht in einem solch kurzen Zeitraum, dass keine Kollisionserkennung schnell genug ist, diese zu erkennen, bzw. die Motoren dieser Kraft entgegen zu richten. Eine solche Erkenntnis stellt den Sinn einer Kollisionserkennung in Frage. Aber zumindest die vom Motor übertragenen Momente können aufgrund der Verzögerung detektiert und mit einer geeigneten Kollisionsstrategie gelindert werden. Das wiederum schont nicht nur die in der Kollision beteiligte Person, sondern auch den Motor (vor allem das Getriebe), der nun solchen Kraftspitzen nicht mehr ausgesetzt ist. Der BioRob hat durch seine Bauweise, was die schnell übertragenen Kräfte angeht, den nachfolgend beschriebenen Vorteil. Dadurch, dass die Motoren nicht in den Gelenken, sondern nahe an der Basis liegen, bewirkt die daraus folgende Massenverteilung eine wesentlich geringere Trägheit in den Gliedern. Zudem sind diese um ein Vielfaches leichter, als die des „DLR-III Leightweight Manipulator“ der bei der Kollision in Bild 2.3 verwendet wurde. Somit kann man davon ausgehen, dass die durch die Glieder übertragene Kraft zwar mindestens genauso schnell abgegeben wird, diese selbst aber wesentlich geringer ist. Der zweite Vorteil besteht in der Elastizität des BioRob. Diese ist ca. 1000-fach höher als der „DLR-III“. Somit müsste die Verzögerung, nach der das Motormoment in der Kollision wirkt, noch wesentlich größer ausfallen. Daher hat die Kollisionserkennung mehr Zeit diese Kraft zu entdecken und auf diese zu reagieren.

Um die Schwere der erlittenen Verletzung zu beurteilen, behalf man sich in [8] mit dem in der Automobilindustrie üblichen „Head Injury Criterion“. Über dieses

Kriterium wird angegeben, mit wie viel Prozent man wahrscheinlich eine ernste Verletzung erleiden wird. Die gemessenen Kollisionen wurden jeweils mit 2m/s Endeffektorgeschwindigkeit und folgenden Robotern durchgeführt: dem Light-Weight-Robot III, KUKA KR3-SI (54kg), KUKA KR6 (235kg) und dem KUKA KR500 (2350kg). Das erstaunliche Ergebnis war, dass bei allen Manipulatoren die Wahrscheinlichkeit nicht mehr als ein Prozent betrug. Um die Verletzungsbeurteilung zu verfeinern, konzentrierte man sich in [6] und [7] auf die Kraft, die notwendig ist, um Frakturen in Gesichts- bzw. Schädelknochen hervorzurufen. Hier kam man zu dem Ergebnis, dass schon moderate Geschwindigkeiten von 0.5 - 1.0 m/s ausreichen um bei allen Knochen, bis auf das Stirnbein[3], Frakturen zu erreichen. Das Stirnbein widersteht Einschlägen von bis zu 2m/s. Zu diesem Zeitpunkt ist noch unklar, ob der BioRob genauso wie die zum Testen verwendeten Manipulatoren die notwendige Kraft aufbringt, um bei ähnlichen Geschwindigkeiten Frakturen hervorzurufen. Das wesentlich geringere Eigengewicht und die Konzentration der Massen um die erste Hauptachse lassen aber erwarten, dass der Manipulator nicht in der Lage ist, ernsthafte Verletzungen hervorzurufen.

Bisher beschränkte sich die Betrachtung der Sicherheit nur auf den Kopf. In [5] wurden jedoch des Weiteren Versuche den Nacken, die Brust und den Arm betreffend durchgeführt. Hier zeigte sich, dass die Kollisionserkennung in der Lage ist, die Kraft des Aufpralls zu verringern. Dies lag vor allem daran, dass Brust und Arme eine natürliche Nachgiebigkeit haben, welche die Kraftübertragung über einen längeren Zeitraum streckt. Viel wichtiger war jedoch, dass durch die Reaktionsstrategien der Endeffektor aus der Kollision gefahren ist, und somit dem betreffenden Menschen ein zusätzliches Sicherheitsgefühl gegeben hat. Des weiteren wurden die Kräfte, welchen der Motor bei der Kollision ausgesetzt ist, untersucht. Man beobachtete, dass bei steifen Industrierobotern schon bei geringen Geschwindigkeiten die maximalen Gelenkdrehmomente für Millisekunden überschritten wurden. Möglichkeiten, diese Belastung der Gelenke zu verringern sind entweder die Gelenksteifheit zu reduzieren (wie beim BioRob der Fall), oder mit einer schnellen Kollisionserkennung auf den Aufprall zu reagieren. Durch eine solche Kollisionserkennung konnten tatsächlich die auftretenden Gelenkdrehmomente verringert werden.

Eine weitere mögliche Situation wurde bis jetzt noch nicht beleuchtet. Bei den vielseitigen Einsatzmöglichkeiten der Manipulatoren kann es vorkommen, dass auch scharfe und spitze Werkzeuge verwendet werden. Vor allem in solchen Fällen ist es besonders wichtig, eine Kollisionserkennung zu haben; denn würden hier die Trajektorien einfach weiter verfolgt, so könnte das schnell schwerwiegende Folgen für die involvierte Person haben.

Abschließend kann man zusammenfassen, dass die Sicherheit des Menschen aber auch die des Roboters, durch eine Kollisionserkennung (mit angemessenen Reaktionsstrategien), erhöht werden kann. Diese sollte schnell und zuverlässig reagieren, vor allem um vermeintlich gefährliche Situationen schnell aufzulösen und der betroffenen Person wieder ein sicheres Gefühl zu geben. Zudem kann Gewichtsreduzierung und Erhöhung der Elastizität ein Vorteil bei der Kollisionserkennung darstellen.

[3] Der stabilste Knochen am Menschlichen Schädel ist das Strinbein.

Doch wie erreicht man nun eine solche Kollisionserkennung? Bei Industrierobotern wird aufgrund der Sicherheitsbereiche, in denen sich niemand aufhalten darf, meist auf eine solche Erkennung verzichtet. In Einzelfällen (z.B. beim KUKA KR3-SI), findet man ein zusätzliches Bauelement direkt hinter dem Handgelenk, welches beim Aufprall des Endeffektors diesen abknicken lässt und einen sofortigen Stopp veranlasst. Eine weitere Möglichkeit ist, den Endeffektor oder das dort montierte Werkzeug mit Drucksensoren auszustatten. Melden diese dann ein Signal, welches momentan nicht erwartet wird, erfolgt wieder eine sofortige Abschaltung des Roboters. Solche Techniken kommen jedoch beim BioRob nicht in Frage, da diese nicht ein Höchstmaß an Sicherheit gewährleisten können. Zwar wird eine sehr schnelle Reaktion auf einen Aufprall erreicht, aber dies gilt nur für den Endeffektor. Kollisionen an anderen Stellen des Manipulators können hingegen nicht erkannt werden. Zudem benötigt man zusätzliche Bauelemente und Sensoren, die wiederum höhere Kosten verursachen.

Eine Möglichkeit für bahntreue[4] Manipulatoren, bei denen es möglich ist, die Gelenkposition, -geschwindigkeit und -beschleunigung über Sensoren zu ermitteln, liegt in folgendem Ansatz. Die auftretenden Gelenkmomente können in diesem Fall mit denen vergleichen werden, die vom Robotermodell simuliert wurden (folgende Roboterdynamikgleichung aus [2]):

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \tag{2.7}$$

$$\hat{\boldsymbol{\tau}} = \boldsymbol{M}(\boldsymbol{q_d})\ddot{\boldsymbol{q_d}} + \boldsymbol{C}(\boldsymbol{q_d}, \dot{\boldsymbol{q_d}})\dot{\boldsymbol{q_d}} + \boldsymbol{G}(\boldsymbol{q_d}) + \boldsymbol{F}(\boldsymbol{q_d}, \dot{\boldsymbol{q_d}})\dot{\boldsymbol{q_d}} \tag{2.8}$$

Hierbei ist $\boldsymbol{\tau}$ das gemessenen Gelenkmoment, sowie $\hat{\boldsymbol{\tau}}$ das zu diesem Zeitpunkt erwartete Gelenkmoment. Zieht man diese Momente nun voneinander ab, so erhält man die in den Gelenken wirkenden Störmomente:

$$\boldsymbol{\tau_{ext}} \approx \boldsymbol{\tau} - \hat{\boldsymbol{\tau}} \tag{2.9}$$

Nun erkennt man auch, warum es wichtig ist, dass der Manipulator bahntreu ist. Dadurch wird gewährleistet, dass die durch die Regelabweichung zu erwartende Differenz $\boldsymbol{\tau_{ext}}$ minimal ist, und der Fehler (die von extern wirkende Kraft) so exakt wie möglich durch $\boldsymbol{\tau_{ext}}$ ausgedrückt wird. Zeigen sich bei $\boldsymbol{\tau_{ext}}$ schnelle Übergänge, können diese von einer Kollision verursacht worden sein. Damit das Ergebnis robust gegenüber dem Sensorrauschen ist, können zusätzlich die Sensorwerte oder $\boldsymbol{\tau_{ext}}$ geglättet werden.

Dieser Ansatz stellt eine gute grundlegende Überlegung dar, welche für den BioRob leider nicht in Frage kommt. Zum Lösen der inversen Dynamik((2.7), (2.8)) stehen nur die Gelenkpositionen zur Verfügung. Das Montieren weiterer Sensoren und die damit verbundenen Kabel sind am BioRob konstruktionsbedingt nicht möglich. Dieses Problem kann zwar durch einen Bus gelöst werden, der jedoch Steuerelektronik an den Sensoren erfordert und zusätzliches Gewicht mit sich bringt. Diese Elektronik sowie marktübliche Sensoren sind jedoch nicht für Temperaturen, wie sie bei Kryoanwendungen vorkommen, gedacht. Das Lösen der inversen Dynamik ohne die Gelenkgeschwindigkeit und Gelenkbeschleunigung ist jedoch nicht möglich. Um die fehlenden Werte abzuschätzen, bedient man sich der numerischen Differentiation der Gelenkposition. Diese Abschätzung von Gelenkgeschwindigkeit und Gelenkbeschleunigung

[4] Bahntreue Manipulatoren fahren die gegebene Trajektorie so genau wie möglich ab.

kann das Rauschen der Sensoren massiv verstärken, sodass nach dem Berechnen von $\boldsymbol{\tau}$ mit den verrauschten Werten die Entscheidung, ob nun eine Kollision vorliegt oder nicht, ein nicht zuverlässig lösbares Problem darstellt.

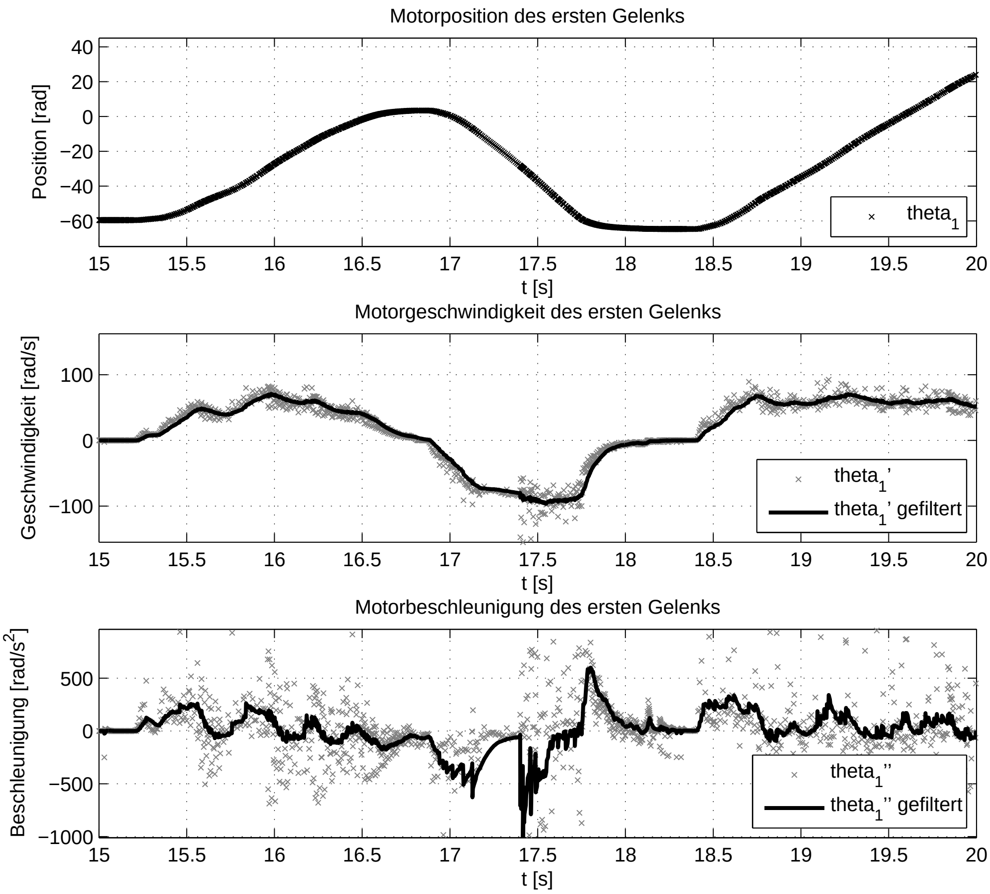


Bild 2.4.: Ermitteln der Motorbeschleunigung über numerische Differentiation

Ein Beispiel für das Verstärken des Rauschens in Sensordaten ist in Bild 2.4 zu sehen. Hier beschreibt der erste Plot die gemessene Motorposition des ersten Gelenks. Im zweiten Plot wird das Ergebnis der numerischen Differentiation durch die Kreuze dargestellt. Hier ist zu erkennen, dass jetzt schon stellenweise Rauschen im Ergebnis auftritt. Um dieses Rauschen zu verringern, liefert ein darauf angewendeter Filter die schwarze Linie. Aber nun besteht der Nachteil, dass diese Linie stellenweise leicht verzögert zu den eigentlichen Daten verläuft. In diesem Falle liegt dies aber noch im Rahmen. Schließlich erhalten wir nach weiterem Differenzieren die Gelenkbeschleunigung im dritten Plot von Bild 2.4. Wie auch im vorherigen Fall tritt wieder Rauschen auf, doch nun in einer weitaus ausgeprägteren Form. Das Ergebnis besteht fast nur noch aus kaum zusammenhängenden Punktwolken. Nach Glättung dieser Beschleunigungsdaten erhält man ein Signal, was leider immer noch verrauscht ist und zudem nicht der Wahrheit entspricht. Betrachtet man sich das Intervall zwischen den Zeitpunkten 17.25s und 17.75s, so fällt auf, dass die ermittelte Beschleunigung zeitweise zu negativ für die Geschwindigkeit ist. Zur Veranschaulichung kann man sich als erste Näherung in diesem Zeitintervall eine lineare Interpolation der Beschleunigung durch eine Gerade vorstellen. Mit solchen unzuverlässigen Werten ist es nicht möglich, ein korrektes τ zu ermitteln. Es besteht zwar die Möglichkeit, den Fehler, welcher durch den Filter verursacht wird, aus dem Ergebnis heraus zu rechnen, jedoch hat auch dies wieder Rauschen zur Folge. Demnach ist die Aufgabe, die Beschleunigung nur mit Hilfe von Positionsdaten zu ermitteln, nicht trivial. Daher ist das Verwenden eines Verfahrens, welches ohne diesen Schritt auskommt ein Schritt in Richtung eines robusten Verfahrens.

Das aktuell implementierte Verfahren zur Kollisionserkennung bedient sich ebenfalls des Vergleichs zwischen einem Soll- und Istwert. Dabei wird jedoch nicht die Veränderung oder Abweichung der Gelenkmomente betrachtet, sondern das Verhalten des Motors. Da die Motorgeschwindigkeit proportional zur angelegten Motorspannug verläuft, ist eine Abschätzung möglich, welche Geschwindigkeit vom Motor erzeugt werden müsste, wenn eine bestimmte Spannung an diesen angelegt ist. Verändert sich diese Geschwindigkeit, bei gleichbleibender angelegter Spannung, so kann dies von einer Kollision verursacht worden sein. Allerdings besteht das Problem, dass nicht nur Kollisionen unerwartete Veränderungen der Motorgeschwindigkeit verursachen können. Auch dynamische Effekte wie die Gravitationskraft oder durch Bewegung wirkenden Corioliskräfte können solch ein Verhalten hervorrufen. Daher besteht die schwierige Aufgabe darin, einen Schwellwert festzulegen, welcher Veränderungen der Motorgeschwindigkeit durch dynamische Effekte zulässt, aber trotzdem bei Kollisionen schnell reagiert. Nach der bisherigen Erfahrung ist es jedoch nicht möglich, einen solchen Schwellwert, der den dynamischen Effekten gegenüber robust Kollisionen erkennt, zu ermitteln. Ein weiteres Manko besteht darin, dass auch bei erfolgreicher Fehlererkennung die wirkenden Gelenkmomente über die gemessene Motorgeschwindigkeit und Gelenkposition nicht zu ermitteln sind. Das bedeutet wiederum, dass nicht festgestellt werden kann, in welchen Gelenken das Moment vom Sollwert abweicht und somit, welches Glied bei der Kollision beteiligt ist. Dazu benötigt man zusätzlich wieder die Gelenkgeschwindigkeit und -position. Ein Ausweg hieraus könnte sein, Kraftsensoren in den Manipulator einzubauen. Aber dann würde man wieder zu dem oben geschilderten Problem, der zusätzlichen Kabeln oder dem Verwenden eines Busses kommen. Aus diesem Grund, sucht man ein Verfahren, welches es ermöglicht, zuverlässig mit den gegebenen Sensoren Kollisionen zu erkennen.

Alle bisher vorgestellten Verfahren haben den Nachteil, dass sie nicht zuverlässig Fehler zwischen den angesteuerten und tatsächlichen Gelenkmomenten ermitteln können. Diese Aufgabe wird bei komplexen dynamischen Anlagen, wie es Roboter darstellen, als „Fault Detection and Isolation“ (FDI) bezeichnet. Die Fehlerermittlung besteht daraus, ein Diagnosesignal, welches sich auf die gesuchten Fehler bezieht, zu generieren. Dieses Signal wird als Residuum bezeichnet. Über den gesamten Zeitraum der Ausführung wird aus dem Verlauf der Eingangsgröße und der Ausgangsgröße ein Residuum berechnet.

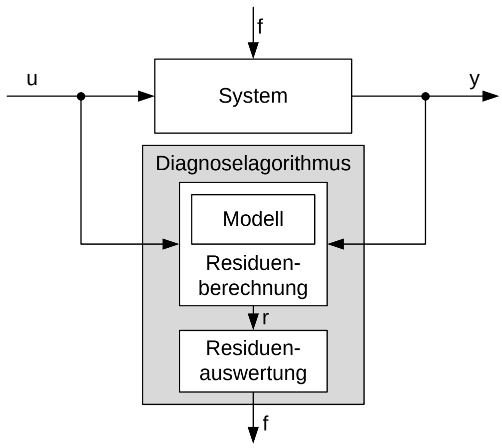


Bild 2.5.: Schematischer Aufbau zur Ermittlung/Auswertung eines Residuums aus [12] (Steuervariable u, Gemessener Systemausgang y, Residuum r, Störung f)

Unter einem Residuum versteht man also eine Größe, die die Abweichung des gemessenen Prozessverhaltens vom Verhalten eines Modells anzeigt. Von einem Residuum erwartet man, dass es sich wesentlich von Null unterscheidet, wenn ein Fehler auftritt. Wird eine vorgegebener Grenzwert vom Residuum überschritten, so kann man davon ausgehen, dass ein Fehler vorliegt.

Ein FDI-Verfahren, welches sich auf die Gelenkmomente von Robotermanipulatoren bezieht, wurde in [4] vorgestellt. Dieses sich auf den verallgemeinerten Impuls des Roboters stützende Verfahren hat gerade genau die Eigenschaft, keine Abschätzung der Gelenkbeschleunigung zu benötigen. Zudem glättet es die gemessenen Werte, sodass auch Rauschen, welches von den Sensoren ausgeht, keinen großen Einfluss auf die Robustheit des Verfahrens hat. Deshalb sollte ein so erzeugtes Residuum das Isolieren von Fehlern zuverlässig ermöglichen, sodass mit angemessenen Reaktionsstrategien (wie in [3] vorgestellt) auf die Kollision reagiert werden kann. Insbesondere für uns hat dieses Verfahren eine große Bedeutung, da es so allgemein ist, dass ohne Probleme auch hohe Gelenkelastizitäten in das Vorgehen eingebunden werden können. Aus diesen Gründen ist dies das Verfahren unserer Wahl um die Kollisionserkennung zu verwirklichen. Weitere Details zu diesem Verfahren werden im Kapitel 3.2 erläutert.

3. Kollisionserkennung und Reaktionsstrategien

3.1. Modellierung des gelenkelastischen Manipulators BioRob

Im folgenden Abschnitt soll zunächst der gelenkelastische Manipulator BioRob vorgestellt werden [11]. Dazu gehören das für die Simulation notwendige Modell, sowie die analytischen Gleichungen. Zu Beginn der Arbeit war ein SimMechanics Modell des Manipulators in Matlab Simulink vorhanden. Dieses war in [1] entwickelt und anschließend in weiteren Arbeiten erweitert worden.

In Rahmen dieser Arbeit wurden das Simulink-Modell und dessen Bibliothek weiter ausgebaut und mit Blöcken zur Kollisionserkennung, Kollisionsbehandlung und mit dem analytischen Dynamikmodell des viergelenkigen Manipulators ergänzt. Mit den analytischen Dynamikgleichungen war eine Validierung des SimMechanics Modells möglich.

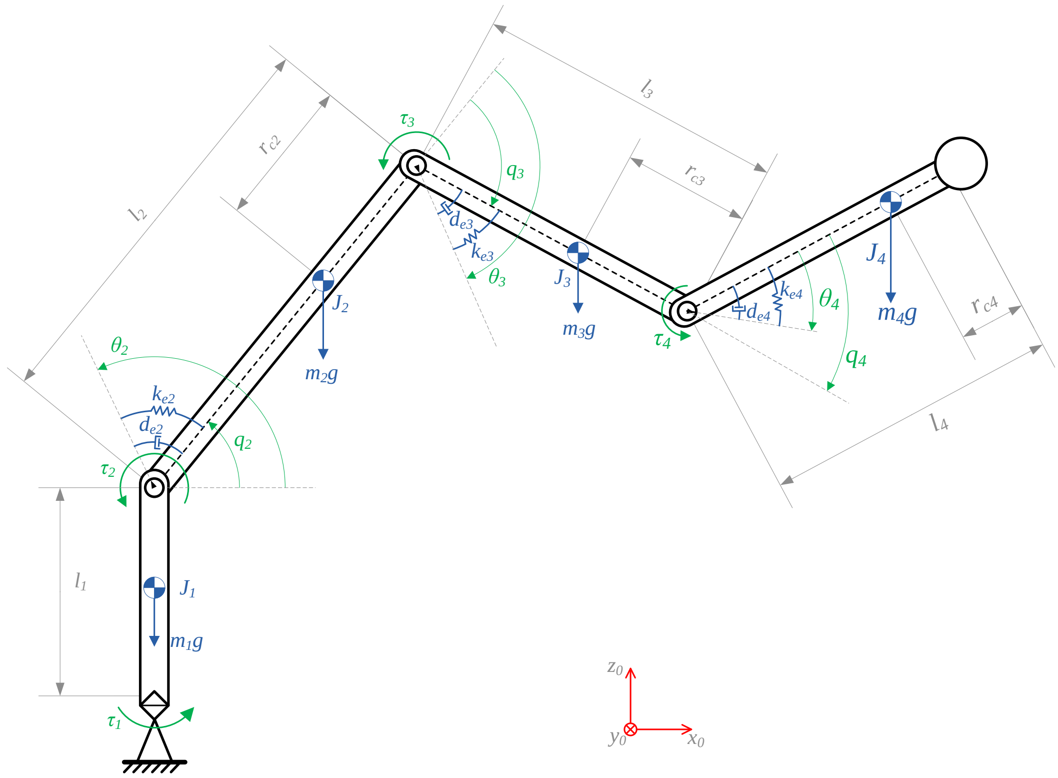


Bild 3.1.: 4 DoF Manipulator mit elastischen Gelenken

In Bild 3.1 ist der schematische Aufbau des BioRob zu sehen. Hierbei können folgende Annahmen getroffen werden: Die Glieder bestehen aus dünnen Stäben mit Masse m_i und Trägheit J_i um den Schwerpunkt. Die Lage des Schwerpunkts wird dabei über die Länge r_{ci}, den Abstand des Schwerpunkts zu S_i im Gelenk $i+1$ bestimmt und ergibt sich aus der Subtraktion von l_i und r_{ci}. Ein Vorteil der Bauweise des BioRob besteht darin, dass die Elastizität, welche über die Seilzüge auftritt, recht einfach aus linearer Gelenkelastizität (lineare Tellerfeder mit viskoser Dämpfung) modelliert werden kann. Die wirkende Federkonstante wird durch k_{ei} und die Dämpfung durch d_{ei} beschrieben. In den Gliedern wird keine weitere Elastizität angenommen. Schließlich gilt es noch, die Reibung der Gelenke zu berücksichtigen. Die Gelenke sind so konstruiert, dass man sie mit einer viskosen Dämpfung, definiert durch d_i in den Gelenken annehmen kann. Die Motoren besitzen das Trägheitsmoment J_{m_i} und werden als ideal spannungsgeregelt angenommen. Wird im Systemeingang keine Spannung angelegt, kann der Motor bei Belastung keine Bremswirkung erzeugen.

3.1.1. Modellierung eines elastischen Gelenks

Bevor das gesamte analytische Modell des Roboters vorgestellt wird, ist es wichtig zunächst einmal zu verstehen, wie sich das Modell für ein Gelenk darstellt. Aus diesem Modell kann dann das Gesamtmodell erschlossen werden.

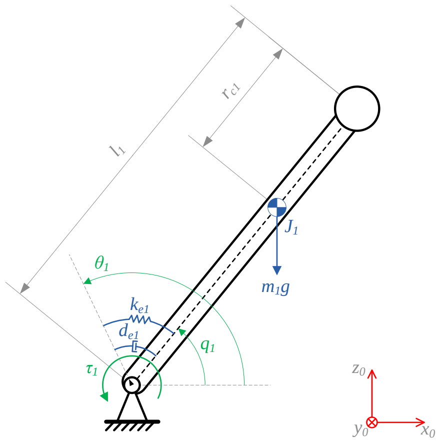


Bild 3.2.: Modell eines elastischen Gelenks

Im Gegensatz zur Modellierung von Robotern mit starren Gliedern, hat man im elastischen Fall zwei Gelenkkoordinaten. Wie in Bild 3.2 aufgeführt, soll $\boldsymbol{\theta} \in \mathbb{R}^N$ die Motorpositionen (nach dem Einrechnen der Getriebeübersetzung) und $\boldsymbol{q} \in \mathbb{R}^N$ die Position der Glieder sein. Außerdem besitzt jedes Gelenk eine lineare Feder, um die elastische Bewegungsübertragung von Motor auf das Gelenk darstellen zu können. Zusätzlich zur Federkraft wirkt jedoch auch noch eine Dämpfung. Diese wird hier im Allgemeinen in ein Modell für die elastische Kopplung eingegliedert. Dieses Feder-Dämpfer Modell ist mit Hilfe zweier Sätze zu erklären.

In Bild 3.3 ist ein Feder-Dämpfer Modell zu sehen, in dem die wirkenden Kräfte in Feder und Dämpfer frei geschnitten sind und in die Kraft F resultieren. Dabei ist

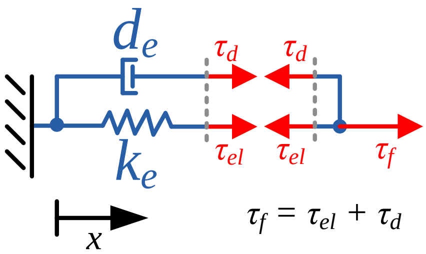


Bild 3.3.: Feder-Dämpfer Modell inkl. Freischnitt der in dem Modell wirkenden Kraft

x die Dehnung, welche auf die Feder und den Dämpfer wirkt. Nach dem Hook'schen Gesetz folgt $\tau_{el} = -k_e \cdot x$ sowie über die viskose Dämpfung $\tau_d = -d_e \cdot \dot{x}$. Somit lautet die Resultierende F: $\tau_f = \tau_{el} + \tau_d = -(k_e \cdot x + d_e \cdot \dot{x})$. Hierbei ist k_e die Federkonstante und d_e die viskose Dämpfung.

Um das Gelenk inkl. Gelenkelastizität zu modellieren benötigt man folgende Gleichungen:

$$J_m \ddot{\theta} = \tau_m \tag{3.1}$$

$$J\ddot{q} + b\dot{q} + mgl\cos(q) = \tau \tag{3.2}$$

$$k_e(\theta - q) + d_e(\dot{\theta} - \dot{q}) = \tau_f \tag{3.3}$$

Gleichung (3.1) beschreibt die mechanische Dynamik des Motors [9] und kann über den Drehimpulssatz hergeleitet werden. Dieser besagt, dass die zeitliche Änderung des Drehimpulses gleich dem Drehmoment ist: $\tau_m = \frac{d\boldsymbol{L}}{dt}$. Da der Drehimpuls $\boldsymbol{L}$ eines starren Körpers (z.B. Motorwelle) das Produkt seines Trägheitstensors mit seiner Winkelgeschwindigkeit ist, also $\boldsymbol{L} = J_m\dot{\theta}$, erhält man durch Einsetzen die Dynamikgleichung des Motors $\tau_m = J_m\ddot{\theta}$.

Gleichung (3.2) steht für die abtriebsseitige Dynamik des Roboters [2] und Gleichung (3.3) steht für das Modell der elastischen Kopplung. Im Feder-Dämpfer Modell wurde die Dehnung/Stauchung x durch die Differenz von Motor- und Gelenkposition ersetzt, die eben genau dies angibt. Wir haben also drei Systeme und deren Modell, die nun miteinander gekoppelt werden müssen, um das Modell des Gelenks zu erhalten. Dazu zunächst einmal die schematische Darstellung der beteiligten Bauteile inklusive der wirkenden Momente.

Die Summe der Kräfte/Momente kann nur in einem Punkt oder Festkörper gebildet werden. Die Kraftübertragung findet vom Motor auf das Gelenk oder umgekehrt statt. Dabei durchläuft die Kraft die, aus keinem Festkörper bestehende, elastische Kopplung. Demnach müssen wir die Angriffspunkte Motor $\leftrightarrow$ Kopplung und Kopplung $\leftrightarrow$ Gelenk betrachten. Hier gilt die in der Mechanik übliche Gleichgewichtsforderung, dass die Summe der Momente Null sein muss. Um die in Bild 3.4 gezeigten Bauteile des Gelenkmodells miteinander zu koppeln, folgen also die zwei Gleichungen:

$$\tau_1 = \tau_f \tag{3.4}$$

$$\tau_2 = \tau_f \tag{3.5}$$

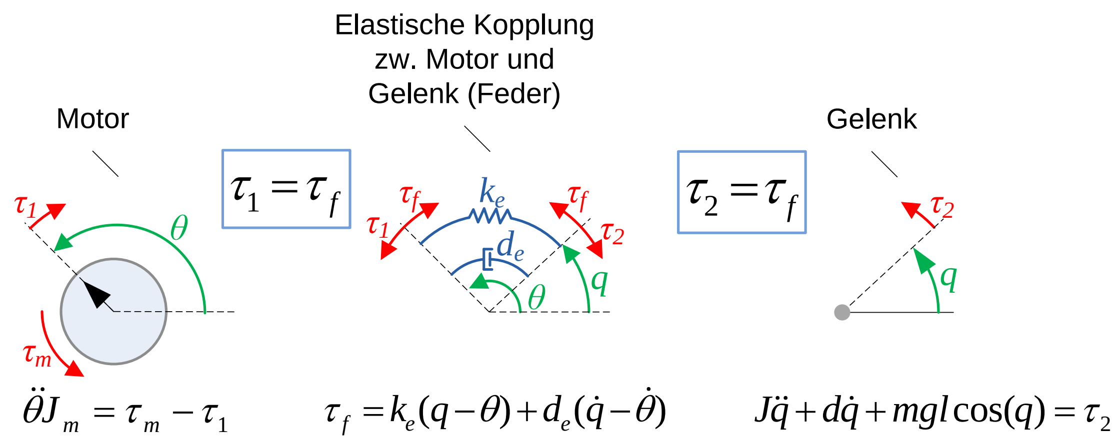


Bild 3.4.: Freigeschnittenes, elastisches Gelenk inkl. der wirkenden Kräfte

Gleichung (3.4) wirkt im Angriffspunkt des Motors, Gleichung (3.5) im Angriffspunkt des Manipulators. Fügt man jetzt τ_1 und τ_2 in die beiden äußeren Systeme (Motor und Manipulator) ein, erhält man die endgültigen Gleichungen des Gelenks:

$$J_m\ddot{\theta} + k_e(\theta - q) + d_e(\dot{\theta} - \dot{q}) = \tau_m \tag{3.6}$$

$$J\ddot{q} + b\dot{q} + mgl\cos(q) = k_e(\theta - q) + d_e(\dot{\theta} - \dot{q}) \tag{3.7}$$

3.1.2. Mehrgelenkiges Robotermodell

Dieses eben beschriebene Gelenkmodell muss nun auf n Gelenke erweitert werden. Zunächst muss sichergestellt sein, dass die beschriebenen Gleichungen im Fall mehrerer Freiheitsgrade immer noch korrekt sind. Zunächst betrachten wir die Motorgleichung. Die vorgenommene Modellierung über das Freischneiden hat auch Bestand, wenn es sich nicht nur um feste, sondern auch im Raum bewegte Motoren handelt. Wie beim Lagrange-Ansatz kann man überlegen, dass die kinetische Energie des Motors vor allem durch seine eigene Rotation verursacht wird. Durch die beim BioRob vorliegenden hohen Motorübersetzungen (ca. $i = 100 : 1$), dreht sich der Motor relativ zum Gelenk sehr schnell. Im Term der kinetischen Energie kann man dann die Bewegung der Motoren im Raum gegenüber der Rotation des Motors um seine eigene Achse vernachlässigen. Also kann das angegebene Motormodell für jedes der n Glieder angenommen und in Matrixschreibweise formuliert werden (siehe unten).

Da für die Gelenkelastizität das Feder-Dämpfer Modell nur angenommen wird, jedoch nicht als solches im Gelenk vorliegt, also es keine Masse hat, wird auch dieses durch die Bewegung der Gelenke im Raum nicht beeinflusst. Somit kann die elastische Kopplung in allen Gelenken weiterhin durch dieses Feder-Dämpfer Modell beschrieben werden.

Als letzten Bestandteil des Modells muss noch die Roboterdynamik betrachtet werden. Die oben hierfür angeführte Gleichung beschreibt die Dynamik eines 1DoF. Um nun das Modell eines n-DoF zu erhalten, transformieren wir die Dynamikgleichung in Matrixschreibweise [2]. Insgesamt lautet das Dynamikmodell also:

$$\boldsymbol{B}\ddot{\boldsymbol{\theta}} + \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}) = \boldsymbol{\tau_m} \tag{3.8}$$

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}) \tag{3.9}$$

wobei $\boldsymbol{M}(\boldsymbol{q})$ die Massenmatrix des Roboters ist, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ die Zentrifugal- und Corioliskräfte angibt, $\boldsymbol{G}(\boldsymbol{q})$ die Gravitationskräfte und $\boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ die Reibung/Dämpfung der widerspiegeln. Der Term $\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}})$ stellt die elastische Kopplung der Gelenke mit dem Motor dar, wobei $\boldsymbol{K}$ die positiv definite, Diagonalmatrix der Gelenksteifheit (Federkonstanten) sowie $\boldsymbol{D} \geq 0$ die Diagonalmatrix der Gelenkviskosität ist. $\boldsymbol{B}$ beschreibt die ebenfalls positiv definite Diagonalmatrix der Motorträgheiten. Da hier nun das allgemeine Modell für einen n-DoF dargestellt ist, sind natürlich die Gelenk- und Motorpositionen $\boldsymbol{q}$, $\boldsymbol{\theta}$ n-dimensionale Vektoren, in denen jeder Eintrag für die Position in einem Gelenk beziehungsweise Motor steht.

3.2. Modellbasierte Kollisionserkennung des gelenkelastischen Manipulators

Nachdem das Dynamikmodell für den BioRob erläutert wurde, liegt der nächste Schritt darin, eine an diesem aufgetretene Kollision zu identifizieren. Wie in Kapitel 2 schon beschrieben, benötigen wir ein sogenanntes Residuum, welches uns ermöglicht, Fehler im System (bei uns eine Kollision) zu erkennen. Wie in [3] bedienen wir uns dabei des verallgemeinerten Impulses. Der generalisierte Impuls kennzeichnet zusammen mit dem Ort den jeweiligen Zustand eines Systems, der sich mit der Zeit gemäß der Hamiltonschen Bewegungsgleichungen ändert. Beim Roboter ist der generalisierte Impuls gleich dem Impuls aus der Newtonschen Mechanik:

$$\boldsymbol{p} = \boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.10}$$

Nach dem Ableiten dieser Gleichung erhalten wir die Dynamikgleichung, welche die Änderungen des Impulses im Manipulator angibt.

$$\dot{\boldsymbol{p}} = \frac{d}{dt}(\boldsymbol{M}(\boldsymbol{q}))\dot{\boldsymbol{q}} + \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} \tag{3.11}$$

Im normalen Betrieb des Manipulators lautet die Gleichung für die Roboterglieder wie in (3.9) gezeigt $\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}})$. Da wir das System beurteilen müssen, wenn eine Kollision aufgetreten ist, benötigen wir auch die entsprechende Dynamikgleichung. In dieser kommt zu dem Moment, das über die elastische Kopplung vom Motor an das Gelenk weiter gegeben wird, noch ein Störmoment, welches von außen auf den Manipulator einwirkt. Also ergibt sich die Dynamikgleichung $\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} = \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}) + \boldsymbol{\tau_k}$. Dieses System kann nun folgendermaßen umgestellt werden:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} = \underbrace{\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}})}_{\boldsymbol{\tau_f}} + \boldsymbol{\tau_k} - \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}. \tag{3.12}$$

Vorbereitend für den nächsten Schritt wird nun noch eine Eigenschaft der Roboterdynamikgleichung ausgenutzt. Die schiefsymmetrische Eigenschaft der Massenmatrix beschreibt die Beziehung zwischen der Massenmatrix $\boldsymbol{M}(\boldsymbol{q})$ und die über die Christoffel-Symbole definierte Coriolis-Matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$. Der mit diesen Matrizen berechnete Term $\dot{\boldsymbol{D}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ ist schiefsymmetrisch. Aus dieser Eigenschaft ergibt sich folgende Gleichung [3]:

$$\dot{\boldsymbol{M}}(\boldsymbol{q}) = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})^T \tag{3.13}$$

Da nicht sofort ersichtlich ist, dass (3.13) gilt, folgt zunächst der Beweis (dieser ist in den Standardlehrwerken nicht auffindbar). Wie bekannt, bezeichnet $\boldsymbol{M}(\boldsymbol{q})$ die Massenmatrix des Manipulators. Aus [14] wissen wir, dass die (k,j)-te Komponente von $\dot{\boldsymbol{M}}(\boldsymbol{q})$ mit Hilfe des totalen Differentials als

$$\dot{m}_{kj} = \sum_{i=1}^{n} \frac{\partial m_{kj}}{\partial q_i} \dot{q}_i \tag{3.14}$$

dargestellt werden kann (diese Gleichung gilt allgemein für Funktionen mit mehreren Veränderlichen, man kann sie aus dem vollständigen Differential herleiten).

Ebenfalls wie in [14] beschrieben, wird die Coriolismatrix über die Christoffel-Symbole definiert. Somit lautet deren (k,j)-tes Element

$$c_{kj} = \sum_{i=1}^{n} \frac{1}{2} (\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_j} - \frac{\partial m_{ij}}{\partial q_k}) \dot{q}_i \tag{3.15}$$

Nun können wir den Beweis führen:

$$\begin{aligned}
\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})^T &= c_{kj} + c_{jk} \\
&= \sum_{i=1}^{n} \frac{1}{2} (\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_j} - \frac{\partial m_{ij}}{\partial q_k}) \dot{q}_i + \sum_{i=1}^{n} \frac{1}{2} (\frac{\partial m_{jk}}{\partial q_i} + \frac{\partial m_{ji}}{\partial q_k} - \frac{\partial m_{ik}}{\partial q_j}) \dot{q}_i \\
&= \sum_{i=1}^{n} \frac{1}{2} (\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{ki}}{\partial q_j} - \frac{\partial m_{ij}}{\partial q_k} + \frac{\partial m_{jk}}{\partial q_i} + \frac{\partial m_{ji}}{\partial q_k} - \frac{\partial m_{ik}}{\partial q_j}) \dot{q}_i
\end{aligned}$$

Da $\boldsymbol{M}(\boldsymbol{q})$ symmetrisch ist, folgt dass $m_{ki} = m_{ik}$, $m_{ij} = m_{ji}$ bzw. $m_{kj} = m_{jk}$ gilt. Somit können folgende Vereinfachungen durchgeführt werden:

$$\begin{aligned}
\Leftrightarrow \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})^T &= \sum_{i=1}^{n} \frac{1}{2} (\frac{\partial m_{kj}}{\partial q_i} + \frac{\partial m_{jk}}{\partial q_i}) \dot{q}_i \\
&= \sum_{i=1}^{n} \frac{\partial m_{kj}}{\partial q_i} \dot{q}_i = \dot{m}_{kj} = \dot{\boldsymbol{M}}(\boldsymbol{q})
\end{aligned}$$

Wird nun in (3.11) die Formeln (3.12) und (3.13) eingesetzt, erhalten wir:

$$
\begin{aligned}
\dot{\boldsymbol{p}} &= (\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T)\dot{\boldsymbol{q}} + \boldsymbol{\tau_f} + \boldsymbol{\tau_k} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \\
&= \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} + \boldsymbol{\tau_f} + \boldsymbol{\tau_k} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \\
&= \boldsymbol{\tau_f} + \boldsymbol{\tau_k} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} \\
&= \boldsymbol{\tau_f} + \boldsymbol{\tau_k} - \underbrace{(-\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}})}_{\boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}})}
\end{aligned}
$$

$$\Rightarrow \dot{\boldsymbol{p}} = \boldsymbol{\tau_f} + \boldsymbol{\tau_k} - \boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{3.16}$$

Stellen wir diese Formel noch nach $\boldsymbol{\tau_k}$ um, so haben wir die Möglichkeit durch externe Störungen in den Gelenken verursachte Drehmomente zu bestimmen, ohne durch mehrfaches Differenzieren der gemessenen Gelenkposition, die Gelenkbeschleunigung ermitteln zu müssen. Die numerische Berechnung der Gelenkbeschleunigung würde ein zu starkes Rauschen nach sich ziehen, sodass ein robustes Ermitteln der wirkenden Kräfte nicht möglich wäre. Demnach steht uns nun ein Residuum zur Kollisionserkennung zur Verfügung.

$$\boldsymbol{\tau_k} = \dot{\boldsymbol{p}} - \boldsymbol{\tau_f} + \boldsymbol{\alpha}(\boldsymbol{q},\dot{\boldsymbol{q}}) \tag{3.17}$$

Allerdings kann auch bei dieser Konstruktion das Sensorrauschen während der Messung der Gelenkposition die Genauigkeit des Ergebnisses beeinflussen. Um diesen Einfluss zu vermindern, bedient man sich der Funktionsweise eines Tiefpassfilters. Ein Tiefpassfilter lässt kurze, hohe Peaks (hochfrequentes Rauschen) am Eingang, langsamer und gedämpft auf den Ausgang durch. Somit werden die Peaks breiter und weniger hoch („verschmiert“).

Wie aus der Regelungstechnik bekannt [13], kann man hierfür ein PT-Glied verwenden. Dies ist im Frequenzbereich wie folgt definiert:

$$R_i(s) = \frac{1}{1 + T_i \cdot s} U_i(s) \tag{3.18}$$

Dabei ist $U_i(s)$ die Eingangsgröße, bei uns also die zu filternde Kraft $\tau_{k,i}$ und $R_i(s)$ das gefilterte Signal, welches bei uns das Residuum darstellt, mit dem nun die Kollisionskräfte identifiziert werden können. Der Ausdruck $1/(1 + T_i \cdot s)$ wird als Übertragungsfunktion $G_i(s) = R_i(s)/U_i(s)$ bezeichnet und beschreibt das Verhalten der Filter. T_i wird als Zeitkonstante bezeichnet, welche die Reaktionsgeschwindgkeit des Filters, auf einen Sprung im Eingangssignal, bestimmt. Durch Transformation von (3.18) in den Zeitbereich erhalten wir folgende Differentialgleichung des Tiefpassfilters (Eingang u durch $\tau_{k,i}$ ersetzt).

$$T_i \cdot \dot{r_i} + r_i = \tau_{k,i} \tag{3.19}$$

$$\dot{r_i} = -\frac{1}{T_i} r_i + \frac{1}{T_i} \tau_{k,i}$$

$$\dot{r_i} = -\frac{1}{T_i} r_i + \frac{1}{T_i} \tau_{k,i}. \tag{3.20}$$

Da wir aber nun nicht nur ein Signal, sondern für jede Komponente des Residuums das Signal filtern müssen, ist es erforderlich diese Schreibweise in Matrixschreibweise zu transformieren. Hier ist nun $\boldsymbol{K_I}$ eine Diagonalmatrix, bei welcher auf der Diagonalen die Kehrwerte der Zeitkonstanten $1/T_i$ der einzelnen Filter stehen. Die Bedeutung der Komponenten auf der Diagonalen wird später erläutert. Des weiteren sind $\dot{\boldsymbol{r}}$, $\boldsymbol{r}$ und $\boldsymbol{\tau_k}$ Vektoren, welche eine Anzahl von Elementen, korrespondierend zu der Anzahl an Gelenken, besitzen. Somit erhalten wir die Dynamikgleichung des Residuums:

$$\begin{aligned}\dot{\boldsymbol{r}} &= -\boldsymbol{K_I}\boldsymbol{r} + \boldsymbol{K_I}\boldsymbol{\tau_k} && (3.21)\\ &= \boldsymbol{K_I}(\boldsymbol{\tau_k} - \boldsymbol{r})\\ &= \boldsymbol{K_I}(\dot{\boldsymbol{p}} - \boldsymbol{\tau_f} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{r}) && (3.22)\end{aligned}$$

Der finale Schritt, um $\boldsymbol{r}$ zu erhalten besteht darin (3.22) nach der Zeit zu integrieren. Dies ergibt die Gleichung für das gesuchte Residuum zur Kollisionserkennung:

$$\begin{aligned}\boldsymbol{r} &= \int_0^t (\boldsymbol{K_I}(\dot{\boldsymbol{p}} - \boldsymbol{\tau_f} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{r}))dt\\ &= \boldsymbol{K_I}(\int_0^t (\dot{\boldsymbol{p}} - \boldsymbol{\tau_f} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{r})dt)\\ &= \boldsymbol{K_I}(\boldsymbol{p}(t) + \int_0^t (-\boldsymbol{\tau_f} - \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{r})dt - \boldsymbol{p}(0))\\ &= \boldsymbol{K_I}(\boldsymbol{p}(t) - \int_0^t (\boldsymbol{\tau_f} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{r})dt - \boldsymbol{p}(0))\\ &= \boldsymbol{K_I}(\boldsymbol{p}(t) - \int_0^t (\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}) + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})^T\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) && (3.23)\\ &\quad - \boldsymbol{F}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{r})dt - \boldsymbol{p}(0))\end{aligned}$$

Hierbei ist $\boldsymbol{r}(0) = 0$ als Anfangswert definiert. $\boldsymbol{K_I} > 0$ ist die diagonale Verstärkungsmatrix, $\boldsymbol{p}(t)$ der verallgemeinerte Impuls zum Zeitpunkt $t \geq 0$, wie in (3.10) definiert. Um $\boldsymbol{r}$ zu berechnen, genügt es, die gemessenen Gelenkposition $\boldsymbol{q}$, deren Ableitung sowie auch das angesteuerte Motormoment $\boldsymbol{\tau_m}$ zu kennen. Insbesondere wird keine weitere Differentiation von $\dot{\boldsymbol{q}}$ benötigt, wie in einen der vorgestellten Alternativstrategien noch erforderlich.

3.3. Eigenschaften des Residuums

Es gibt einige wichtige Eigenschaften des eben konstruierten Residuums, die ausschlaggebend dafür sind, dieses Verfahren überhaupt zur Kollisionserkennung nutzen zu können.

Wie gezeigt, genügt die Dynamik von $\boldsymbol{r}$ der Gleichung (3.21). Komponentenweise ist dies Gleichung (3.20). Durch Umformen dieser Gleichung im Frequenzbereich, erhält man so, für jede einzelne Komponente der Dynamik, eine separate Übertragungsfunktion:

$$\frac{r_i(s)}{\tau_{k,i}} = \frac{K_{I,i}}{s + K_{I,i}}, i = 1, \ldots, N. \tag{3.24}$$

So wird die Dynamikgleichung über N entkoppelte Übertragungsfunktionen mit einheitlicher Verstärkung gebildet. Das hierauf aufbauend entwickelte Residuum $\boldsymbol{r}$ besitzt demnach N unabhängige Komponenten.

Für die nächste Eigenschaft wollen wir die Rolle der Zeitkonstante des Filters etwas genauer beleuchten. Dazu betrachten wir uns noch einmal die Differentialgleichung (DGL) des Filters (Gleichung (3.19)). Um die Zeitkonstante für einen Filter zu ermitteln, benötigen wir die Lösung der Differentialgleichung (3.19). Diese setzt sich aus der Lösung der homogenen DGL sowie der inhomogenen DGL zusammen.

Über den Ansatz $r_{i,h}(t) = e^{\lambda t}$ (h im Index steht für homogen) kann man die homogene DGL $T_i \cdot \dot{r}_i + r_i = 0$ lösen:

$$\begin{aligned} 0 &= T_i \cdot \dot{r}_i(t) + r_i(t) \\ 0 &= \lambda e^{\lambda t} T_i + e^{\lambda t} \\ 0 &= (\lambda T_i + 1) e^{\lambda t} \\ 0 &= \lambda T_i + 1 \\ \Rightarrow \lambda &= -\frac{1}{T_i} \end{aligned}$$

Diese Nullstelle (Lösung) in den Ansatz eingesetzt ergibt die Lösung der homogenen DGL: $r_{i,h}(t) = e^{-\frac{1}{T_i}t}$.

Um das Zeitverhalten zu bestimmen genügt es den inhomogenen Fall zu vereinfachen, indem man davon ausgeht, dass die Eingangsgröße konstant ist. Somit kann man leicht bestimmen, nach welcher Zeit sich das Ausgangssignal an das Eingangssignal angepasst hat. Für $\tau_{k,i}(t) = T_i \cdot \dot{r}_i(t) + r_i(t)$ sei also $\tau_{k,i}(t) = \tau = const$. Daraus folgt, dass auch die Lösung der inhomogenen Gleichung konstant sein muss, also $r_{i,i}(t) = const$ (das zusätzliche i im Index steht für inhomogen) gelten muss.

Die Gesamtlösung von $r_i(t)$ ergibt sich aus der homogenen, sowie inhomogenen Lösung: $r_i(t) = C_1 r_{i,h}(t) + C_2 r_{i,i}(t)$. Die ermittelten Lösungen eingesetzt ergeben:

$$r_i(t) = C_1 e^{-\frac{1}{T_i}t} + C_3 \tag{3.25}$$

$$\Rightarrow \dot{r}_i(t) = -\frac{C_1}{T_i} e^{-\frac{1}{T_i}t} \tag{3.26}$$

Schließlich müssen noch die Konstanten der Lösung ermittelt werden. Hierzu setzen wir die ermittelte Lösung (3.25) und deren Ableitung (3.26) in die inhomogene DGL $\tau_{k,i}(t) = T_i \cdot \dot{r}_i(t) + r_i(t)$ ein und erhalten: $\tau_{k,i} = (-\frac{C_1}{T_i} e^{-\frac{1}{T_i}t}) T_i + C_1 e^{-\frac{1}{T_i}t} + C_3$. Da diese DGL für alle t gelten muss, kann man t geschickt wählen um die erste Bedingung, zur Ermittlung der Konstanten, aufzustellen. Wir wählen hierfür $t = 0$, da das System einen Startwert genügen muss, bevor der Sprung auf den gegebenen Eingangswert $\tau_{k,i}$

ausgeführt wird. Demnach ist es möglich voraus zu setzen, dass $r_i(0) = 0$, also das System zum Zeitpunkt Null in Ruhe ist.

Aus diesen zwei Bedingungen erhalten wir ein Gleichungssystem, welches man wie folgt lösen kann. Für $t = 0$ erhalten wir die Gleichung:

$$\begin{aligned} \tau_{k,i} &= -C_1 + C_1 + C_3 \\ &\Rightarrow C_3 = \tau_{k,i} \end{aligned} \tag{3.27}$$

Aus $r_i(0) = 0$ erhalten wie die Gleichung:

$$\begin{aligned} 0 &= C_1 + C_3 \\ &\Rightarrow C_1 = -C_3 \end{aligned} \tag{3.28}$$

Setzt man nun noch Gleichung (3.27) in Gleichung (3.28) ein erhält man $C_1 = -C_3 = -\tau_{k,i}$ und weiter als Endergebnis:

$$\begin{aligned} r_i(t) &= -\tau_{k,i} e^{-\frac{1}{T_i}t} + \tau_{k,i} \\ &= (1 - e^{-\frac{1}{T_i}t})\tau_{k,i} \\ \Rightarrow \dot{r}_i(t) &= \frac{1}{T_i} e^{-\frac{1}{T_i}t} \tau_{k,i} \end{aligned}$$

Nachdem die Lösung für die DGL des Systems (des Filters) ermittelt ist, kann die Zeitkonstante ermittelt werden.

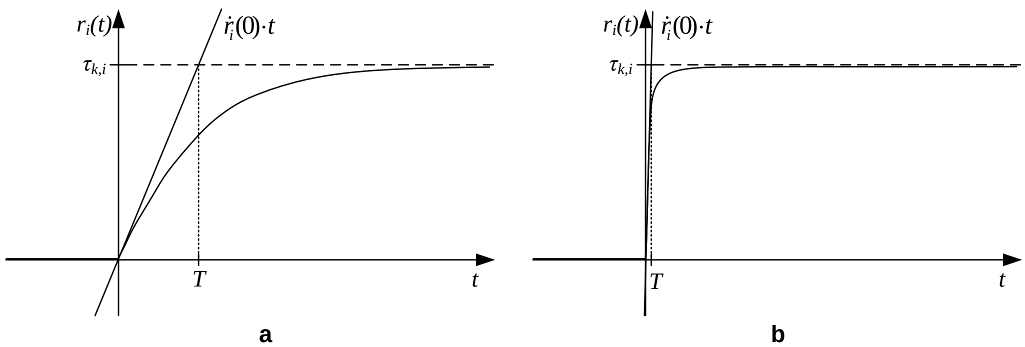


Bild 3.5.: Sprungantwortverhalten des Filters in Abhängigkeit von der Zeitkonstante T

Diese ermittelt sich, wie in Bild 3.5 a gezeigt, über die Ableitung der Lösung der DGL zum Zeitpunkt Null. Diese Steigung, betrachtet über den Zeitverlauf, ergibt eine Gerade. Der Zeitpunkt des Schnittpunkts dieser Geraden mit dem Eingangswert beschreibt den Wert von T. Schiebt man nun T in Richtung Null, vergrößert sich die Anfangssteigung des Filters, wie es in Bild 3.5 b zu sehen ist. Demnach gilt:

$$\boldsymbol{T \longrightarrow 0 \Rightarrow K_I \longrightarrow \infty \Rightarrow r \approx \tau_k}$$

Zusammenfassend kann festgehalten werden, dass T_i für jeden Filter so klein wie möglich gewählt werden muss, damit $\boldsymbol{r} \approx \boldsymbol{\tau_k}$ gilt. Hierbei ist jedoch zu beachten, dass T_i nicht zu klein sein darf, weil sonst wieder kurze Peaks im Eingangssignal fast unverändert vom Filter ausgegeben werden, die erwünschte Glättung also nicht mehr stattfindet. Eine weitere Schwierigkeit liegt darin, dass die Dynamik von r durch $K_{I,i} = \frac{1}{T_i}$ beeinflusst wird. Das Bilden von $\frac{1}{T_i}$ für sehr kleine T_i ist jedoch numerisch ein Problem.

Die letzte Eigenschaft des Residuums bezieht sich darauf, wie es bei einer Kollision aufgebaut ist bzw. wann Kollisionen erkannt werden können. Allgemein kann man unterscheiden zwischen Kollisionen die auftreten, wenn der Manipulator in Bewegung ist, oder solchen, bei denen er still steht. Im statischen Fall ($\dot{\boldsymbol{q}} = 0$) kann über die sogenannte „virtuelle Arbeit“ aus der Mechanik, die Beziehung zwischen der von extern wirkenden Kraft und den in den Gelenken wirkenden Momenten hergestellt werden.

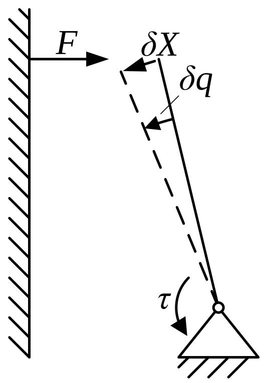


Bild 3.6.: Freikörperbild eines 1DoF, welcher virtuelle Arbeit gegen eine Wand ausübt

Der in Bild 3.6 zu sehende eingelenkige Manipulator steht in Kollision mit der Wand. Dabei kann man $\delta\boldsymbol{X}$ und $\delta\boldsymbol{q}$ als eine infinitesimale Verrückung des Gliedes im Arbeits- bzw. Gelenkraum bezeichnen. Da der Manipulator gegen die Wand drückt und das Glied in Wirklichkeit nicht verrückt wird, bezeichnet man diese Verrückung als virtuelle Verrückung. Die dabei verrichtete Arbeit der Wand gegen den Manipulator bzw. des Gelenks gegen die Wand müssen gleich sein und lauten nach [10]

$$\boldsymbol{\tau}^T \delta\boldsymbol{q} = \boldsymbol{F}^T \delta\boldsymbol{X}. \tag{3.29}$$

Nach [14] stehen die virtuellen Verrückungen des Arbeits- und Gelenksraums über die Jakobimatrix wie folgt in Zusammenhang

$$\delta\boldsymbol{X} = \boldsymbol{J}(\boldsymbol{q})\delta\boldsymbol{q}. \tag{3.30}$$

Setzt man (3.30) in (3.29) ein, erhält man nach einigen Umformungen eine Beziehung zwischen der am Endeffektor wirkenden Kraft und dem dazu passenden Gelenkmoment:

$$\begin{aligned} \boldsymbol{\tau}^T \delta\boldsymbol{q} &= \boldsymbol{F}^T \boldsymbol{J}(\boldsymbol{q})\delta\boldsymbol{q} \\ \boldsymbol{\tau}^T &= \boldsymbol{F}^T \boldsymbol{J}(\boldsymbol{q}) \\ \boldsymbol{\tau} &= \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{F} \end{aligned} \tag{3.31}$$

Mit Hilfe dieser Beziehung ist es nun möglich zu erklären, wie das Residuum im statischen Fall aufgebaut sein muss. Wie oben gezeigt, kann man annehmen dass $\boldsymbol{r} \approx \boldsymbol{\tau_k}$ gilt. Über die Beziehung (3.31) folgt weiterhin $\boldsymbol{r} \approx \boldsymbol{\tau_k} =^0 \boldsymbol{J}_i^T(\boldsymbol{q})\boldsymbol{F_k}$. Bei einer wirkenden Kollisionskraft $\boldsymbol{F_k}$ an Glied i sind die letzten $N-i$ Spalten der Jacobimatrix Null. Dies wird über folgende Beziehung klar. Da uns nur die linearen Kräfte am Endeffektor interessieren, benötigen wir im Folgenden auch nur den linearen Anteil der Jacobimatrix. Dieser Anteil berechnet sich aus:

$$^0\boldsymbol{J}_i(\boldsymbol{q}) = \frac{\partial^0 r_i(\boldsymbol{q})}{\partial \boldsymbol{q}} \tag{3.32}$$

Die ^0r_i aus Formel 3.32 erhält man wiederum aus folgendem Zusammenhang mit den Transformationsmatritzen:

$$^0\boldsymbol{T}_1(q_1) \cdot^1 \boldsymbol{T}_2(q_2) \cdot \ldots \cdot^{i-1} \boldsymbol{T}_i(q_i) =^0 \boldsymbol{T}_i(\boldsymbol{q}) = \begin{pmatrix} ^0\boldsymbol{R}_i(\boldsymbol{q}) & ^0\boldsymbol{r}_i(\boldsymbol{q}) \\ \boldsymbol{0}^T & 1 \end{pmatrix} \tag{3.33}$$

Der Vektor $^0\boldsymbol{r}_i$ ist also nur von den ersten i Gelenkpositionen abhängig. Dies hat beim Bilden der Jacobimatrix zur Folge, dass die letzten $N-i$ Spalten Null sind:

$$^0\boldsymbol{J}_i(\boldsymbol{q}) = [\frac{\partial^0 r_i(\boldsymbol{q})}{\partial q_1}, \frac{\partial^0 r_i(\boldsymbol{q})}{\partial q_2}, \ldots, \frac{\partial^0 r_i(\boldsymbol{q})}{\partial q_i}, 0, \ldots, 0] \tag{3.34}$$

Dies kann man sich veranschaulichen, indem man sich vor Augen führt, dass die wirkende Kraft sich nur auf die Gelenke zurück zur Basis entlang der kinematischen Kette auswirkt. Die eventuellen Gelenkpositions- oder Gelenkgeschwindigkeitsänderungen in den Gelenken zum Endeffektor hin werden nur indirekt über die entstehenden dynamischen Effekte (z.B. die durch die resultierende Bewegung der Gelenke 1 bis i auf die Kollision entstehende Corioliskraft) der Kollision verursacht. Bei einer Kollision am i-ten Glied der kinematischen Kette erhalten wir also einen Residuumsvektor der Art

$$\begin{aligned} \boldsymbol{r} =& [* \ \ldots \ * \ * \quad 0 \quad \ \ldots \ 0]^T. \\ & \qquad\qquad\quad \uparrow \qquad\quad \uparrow \\ & \qquad\qquad i+1 \ \ldots \ N \end{aligned} \tag{3.35}$$

Die ersten i Elemente des Residuenvektors sind verschieden von Null. Somit ist es also möglich eine Kollision im statischen Fall ($\dot{\boldsymbol{q}} = 0$) zu erkennen. Das diese Erscheinung auch im dynamischen Fall gilt, kann man sich an der Dynamik von $\boldsymbol{r}$ in (3.21) veranschaulichen. Bei exaktem Modell des Manipulators wird diese nicht von der Robotergeschwindigkeit oder -beschleunigung beeinflusst. Nur das externe Moment $\boldsymbol{\tau_k}$ hat Einfluss auf das aktuelle Residuum. Daraus kann man schließen, dass sich das dynamische Verhalten kaum vom statischen Verhalten unterscheiden wird. Des weiteren kann man die Struktur in (3.35) mit der Newton-Euler Rekursion vergleichen. Auch hier werden die wirkenden Momente in den Gelenken des Manipulators vom Endeffektor zur Basis hin ermittelt. Dabei sind die Momente in einem Gelenk jeweils vom vorherig berechneten abhängig. Wirkt also im $i+1$-ten Gelenk kein Moment, jedoch im i-ten Glied, so beeinflusst dieses Moment nicht dass Moment des $i+1$-ten Gelenks. Auf Grund der Struktur von $\boldsymbol{r}$ während einer Kollision ist es nicht nur möglich eine Kollision zu erkennen, sondern auch zu detektieren welches Glied an dieser beteiligt ist.

Wie schon im Kapitel 2 beschrieben, wird von einem Residuum erwartet, dass es verschwindet, solange kein Fehler auftritt und sich wesentlich von Null unterscheidet, wenn ein Fehler aufgetreten ist. Da diese Forderung erfüllt ist, kann man sich ebenfalls wieder an der Dynamik von $\boldsymbol{r}$ erklären. So lange kein externes Moment $\boldsymbol{\tau_k}$ auftritt, bleibt $\boldsymbol{r}$ unverändert, entspricht also dem Anfangswert Null. Unterscheidet sich aber bei einer Kollision $\boldsymbol{\tau_k}$ von Null, so wird sich das Residuum mit der Struktur von (3.35), entsprechend der Zeitkonstante diesem Wert annähern. Löst sich die Kollision wieder auf, $\boldsymbol{\tau_k}$ also Null wird, so wird auch $\boldsymbol{r}$ wieder Null. Die Geschwindigkeit, in der $\boldsymbol{r}$ den Wert $\boldsymbol{\tau_k}$, bzw. wieder Null annimmt, wird über die Zeitkonstante bestimmt.

3.4. Reaktionsstrategien

Nachdem es uns nun möglich ist, eine Kollision des Manipulators zu erkennen, ist es notwendig, auf diese angemessen zu reagieren, damit unangenehme Quetschungen oder Einklemmungen aufgelöst werden. Im Folgenden werden die fünf Reaktionsstrategien beschrieben, welche als Antwort auf das Entdecken einer Kollision vom Manipulation ausgeführt werden können. Zur Verdeutlichung der Strategien sind diese an einem kleinen Beispiel bildlich dargestellt.

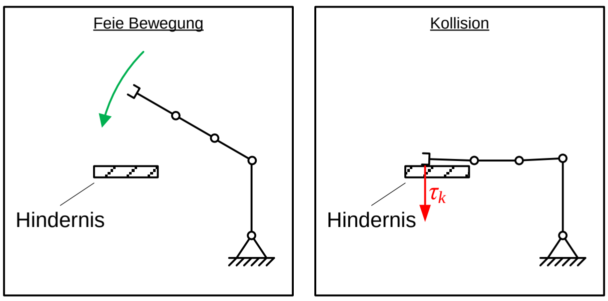


Bild 3.7.: Beispiel einer möglichen Kollision

Der Manipulator führt eine Bewegung in Richtung des grünen Pfeils, wie in Bild 3.7 dargestellt. Beim Durchführen dieser Bewegung stößt der Endeffektor auf ein Hindernis, welches eine Kollision mit dem Moment $\boldsymbol{\tau_k}$ in den Gelenken hervorruft.

Die erste Kollisionsstrategie dient als Referenz, um die Auswirkung einer Kollision ohne Reaktion beurteilen zu können. Daher besteht die erste Strategie darin, nicht zu reagieren. Weil dies keine wirkliche Reaktionsstrategie ist, wird sie als Strategie 0 bezeichnet. Diese hat im Beispiel zur Folge, dass die Motoren weiterhin versuchen die vorgegebene Bahn nach zu fahren, also wird auch nach der Kollisionserkennung aktiv eine Kraft in die Kollision gegeben.

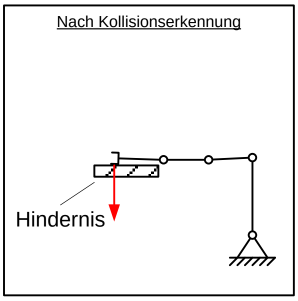


Bild 3.8.: Kollisionsstrategie 0: keine Reaktion, Trajektorie fährt weiter

Als weitere nicht reaktive Strategie wird der Manipulator nach Erkennung der Kollision sofort gestoppt (Strategie 1). Hierbei bleibt der Endeffektor im Hindernis stehen. Da nicht versucht wird, die Bahn weiter zu verfolgen, also keine weitere Kraft von den Motoren in das Hindernis erzeugt wird, ist auch die Kraft im Hindernis geringer, als bei Strategie 0.

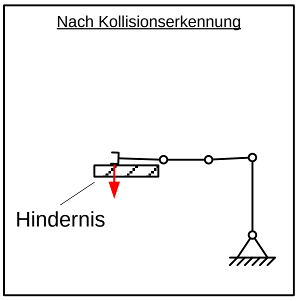


Bild 3.9.: Kollisionsstrategie 1: Trajektorie anhalten

Die kommenden drei Strategien reagieren auf die Kollision, indem die Regelung der Momente, welche auf die Gelenke gegeben werden, durch eine andere Regelung ersetzt wird. In Strategie 2 kommt als Regelung die Gravitationskompensation zum Einsatz. Sobald also die Kollision erkannt wird, schaltet der Positionsregler ab und jede weitere Bewegung wird nur noch gravitationskompensiert durchgeführt. Somit bewegt sich der Manipulator wie schwerelos aus der Kollision. Da jedoch Reibung und die Übersetzung der Motoren in den Gelenken wirkt, wird die Bewegung langsam abgebremst.

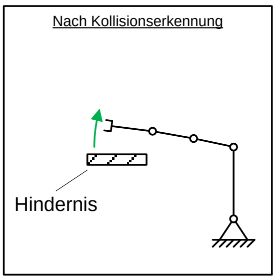


Bild 3.10.: Kollisionsstrategie 2: Positionsregelung deaktivieren, Gravitationskompensation bleibt aktiv

Um eine etwas natürlichere Reaktion des Manipulators auf eine Kollision hervorzurufen, muss das Residuum in die Momentenregelung eingebunden werden. Dieses gibt an, welches Moment in jedem Gelenk wirkt. Diese Momente können genutzt werden, um eben genau in die entgegengesetzte Richtung zur Kollision aus dieser herauszufahren. Sobald dies geschehen ist, wird das Residuum Null. Die Momente in den Gelenken, um aus der Kollision heraus zu fahren, ebenfalls Null, und die weitere Regelung erfolgt wieder gravitationskompensiert. Vorstellen kann man sich das Verhalten wie ein schnelles aus der Kollision heraus fahren mit anschließendem Auslaufen des Arms.

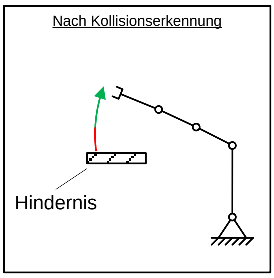


Bild 3.11.: Kollisionsstrategie 3: Das Residuum in die Kraftregelung einbeziehen, um aus der Kollision zu fahren.

In der letzten Strategie wird ähnlich verfahren. Mit Ausnahme, dass nicht das Moment in den Gelenken zum Herausfahren aus der Kollision über das Residuum bestimmt wird, sondern dieses die Geschwindigkeit des Herausfahrens angibt. Dies hat zur Folge, dass nachdem das Residuum null ist, auch die Geschwindigkeit Null ist. Somit erfolgt nach dem schnellen Herausfahren, kein langsames Auslaufen des Manipulators, sondern der Stopp des Arms.

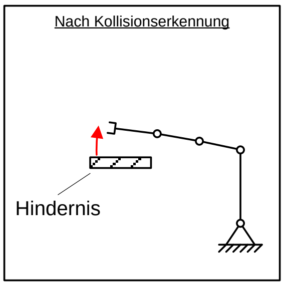


Bild 3.12.: Kollisionsstrategie 4: Das Residuum in die Kraftregelung einbeziehen, um aus der Kollision zu fahren.

Wie genau diese Strategien in die Regelung des Manipulators eingebunden werden, wird im Kapitel 4.1.2 beschrieben.

4. Simulation mit Matlab/Simulink

Bevor das Verfahren zur Kollisionserkennung auf der Hardware des BioRob implementiert werden kann, musste zuvor eine Simulation am Modell des 4-DoF durchgeführt werden. Hierzu diente das Programm Matlab/Simulink. Dieses ermöglicht, mit Hilfe der SimMechanics Toolbox den physikalischen Aufbau des Manipulators nachzubilden. Aus dem so erstellten Modell konnten dann die Werte (Gelenkposition und angelegte Motormomente) entnommen werden, welche im realen Modell über die Sensoren zur Verfügung stehen. Mit diesen Sensorwerten konnte dann das Residuum modelliert und schließlich ausgewertet werden.

In diesem Kapitel wird das Vorgehen für eine möglichst genaue Simulation des Verfahrens skizziert. Des Weiteren folgen einige Simulationen, um das Verhalten des Manipulators während und nach einer Kollision zu beurteilen.

Die Größe und Kraft des BioRob sind vom menschlichen Arm inspiriert. So hat er eine Reichweite von ca. 80 cm und eine Traglast von ca. 0.5 kg (empfohlene Traglast bei Dauerbelastung). Man kann sich leicht vorstellen, dass dies angenehmen Arbeiten eines Menschen entspricht. Der Manipulator selbst hat eine Gewicht von 3.5 kg und besteht aus vier Drehgelenken, welche durch Maxon DC RE30 Gleichstrommotoren angetrieben werden. Das maximale geregelte Drehmoment in den Gelenken ist abtriebsseitig auf 10 Nm beschränkt. Die Umdrehungen der verwendeten Motoren werden über ein Planetengetriebe übersetzt. Die Getriebe haben ein Übersetzungsverhältnis (inkl. der nachgeschalteten Übersetzung durch die Elastizität) von 100:1. Somit ist es möglich, eine maximale Gelenkgeschwindigkeit von ca. $200°/s$ zu erreichen. Dies entspricht einer maximal möglichen linearen Endeffektorgeschwindigkeit von 5.5 m/s.

4.1. Modellierung

Die erste Herausforderung bestand darin, ein möglichst genaues physikalisches Modell vom BioRob zu erstellen. Um dies zu erleichtern, wurden im Laufe der Entwicklung bereits SimMechanics Blöcke erstellt, welche den Antrieb und den Aktuator eines Gelenks nachbildeten. Somit war es möglich, aus n dieser Blöcke den n-DoF zusammen zu stellen. Die detaillierten Spezifikationen wie Position des Massenschwerpunkt im Glied, Motorträgheit, Getriebeübersetzung usw. wurden während der Simulation spezifisch jedem Gelenk und Glied zugewiesen.

Um sicher zu gehen, dass dieses physikalische Modell der Realität sehr nahe kommt, musste das Verhalten des Manipulators mit dem analytischen Modell verglichen werden. Wie schon in Kapitel 3.1.2 lautet das Robotermodell mit elastischen

Aktuatoren

$$\begin{aligned}\boldsymbol{B}\ddot{\boldsymbol{\theta}} + \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}}) &= \boldsymbol{\tau_m} \\ \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} &= \boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}})\end{aligned} \tag{4.1}$$

Um das Modell des elastischen Motors zu erstellen, ist es einfacher, die Modelle der einzelnen Gelenke miteinander zu koppeln, als ein Gesamtmodell zu erstellen. Ein weiterer Vorteil liegt darin, dass so für jede beliebige Anzahl an Gelenken das Motormodell zusammengefügt werden kann. Nachdem das Motormodell erstellt wurde, ist es möglich aus dem angesteuerten Motormoment $\boldsymbol{\tau_m}$ die Federkraft sowie die Dämpfungskraft der elastischen Kopplung $\boldsymbol{K}(\boldsymbol{\theta} - \boldsymbol{q}) + \boldsymbol{D}(\dot{\boldsymbol{\theta}} - \dot{\boldsymbol{q}})$ zu berechnen.

Mit Hilfe des allgemeinen Roboterdynamikmodells und der eben ermittelten elastischen Kopplung ist es möglich, die erwartete Gelenkposition zu ermitteln. Hierfür muss die Gleichung (4.1) nach $\ddot{\boldsymbol{q}}$ umgestellt werden (direktes Dynamikmodell). Durch zweimaliges Integrieren der Gelenkbeschleunigung erhält man dann die Gelenkposition $\boldsymbol{q}$, welche mit der Gelenkposition aus dem SimMechanics-Modell übereinstimmen muss. Der schematische Aufbau des analytischen Modells ist in Bild 4.1 dargestellt.

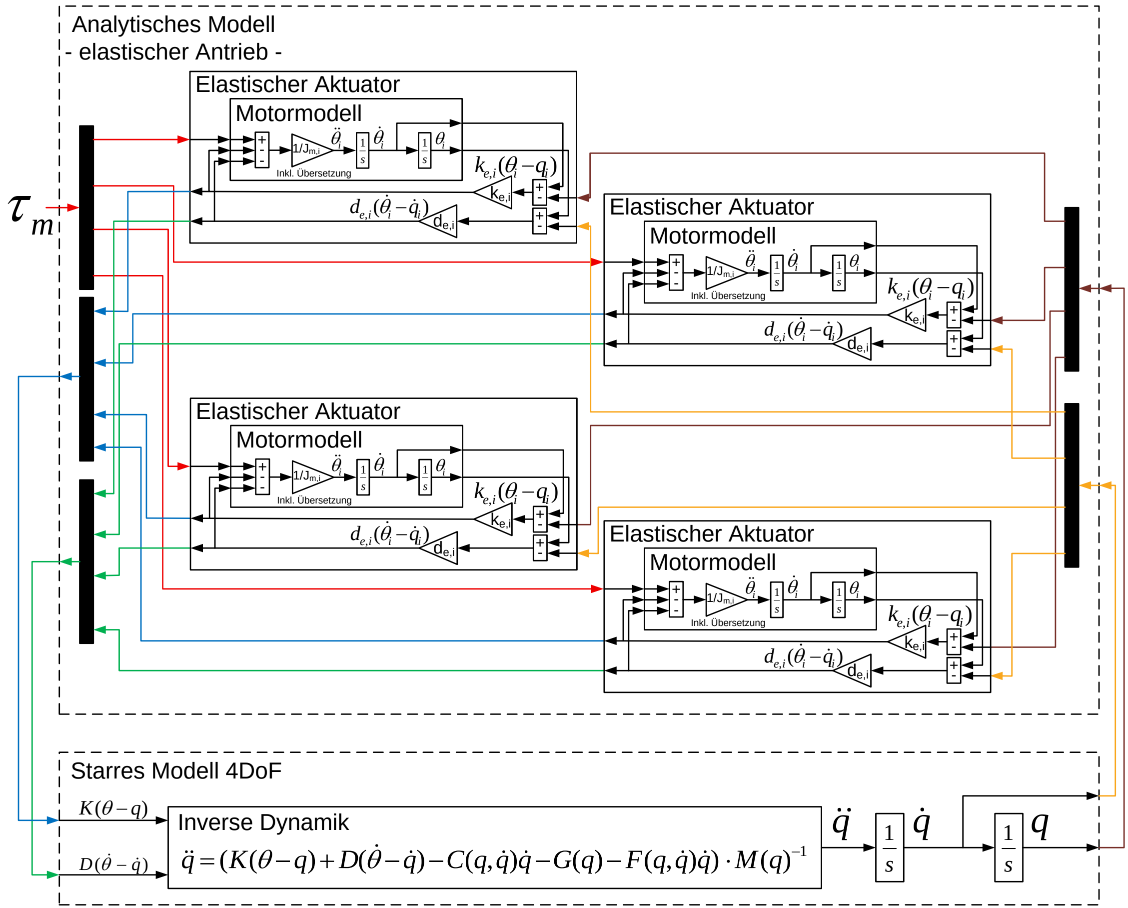


Bild 4.1.: Schematischer Aufbau des 4 DOF Modells

Die Dämpfung der elastischen Kopplung war im bisherigen Modell nicht ausreichend modelliert. Dies wurde korrigiert, um eine bessere Übereinstimmung des simulierten mit dem realen Verhalten des Roboters zu erhalten.

Zur Berechnung der inversen Dynamik sowie zum Berechnen des Residuum ist es notwendig, die Massenmatrix $\boldsymbol{M}(\boldsymbol{q})$, die Matrix der Corioliskräfte $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, sowie die Matrix der Gravitationskräfte $\boldsymbol{G}(\boldsymbol{q})$ aufzustellen. Mit Hilfe des Mathematikprogramms „Mathematica“ [15] sowie des Addons „Robotica“ konnte dies zuverlässig bewältigt werden. Somit war es möglich, nach Angabe der Denavit-Hartenberg (DH) Parameter die gesuchten Matrizen zu ermitteln. Um dies nachvollziehen zu können folgen in Tabelle 4.1 die DH-Parameter sowie in Bild 4.2 die Abbildung des BioRob in der horizontalen Nullstellung passend zu den DH-Parametern.

i	θ_i	d_i	a_i	α_i
1	q_1	l_1	0	$\frac{\pi}{2}$
2	q_2	0	l_2	π
3	q_3	0	l_3	0
4	q_4	0	l_4	0

Tabelle 4.1.: DH-Parameter des BioRob

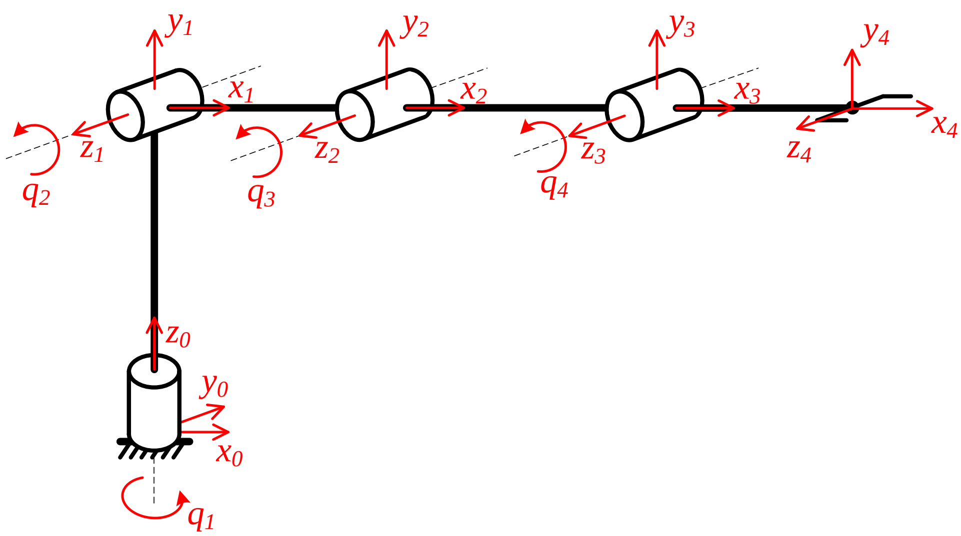


Bild 4.2.: Horizontale Nullstellung des BioRob zu den gegebenen DH-Parametern

Die Matrizen aus der Mathematicaausgabe können jedoch noch nicht in der Matlabsimulation verwendet werden. Dazu muss noch die Syntax angepasst werden. Die schließlich verwendeten Matrizen sind im Anhang zu finden. Die letzte noch verwendete Matrix der Reibung (viskose Dämpfung in den Gelenken) $\boldsymbol{F}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ muss nicht berechnet werden, da uns diese Werte aus einer früheren Arbeit zur Verfügung stehen.

4.1.1. Modellierung des Residuums

Aus den Dynamikgleichungen kann schließlich, nach Gleichung (3.23), die Gleichung für das Residuum ermittelt werden. Eine Möglichkeit, diese Gleichung in Matlab zu implementieren, ist im folgenden Blockdiagramm (Bild 4.3) dargestellt.

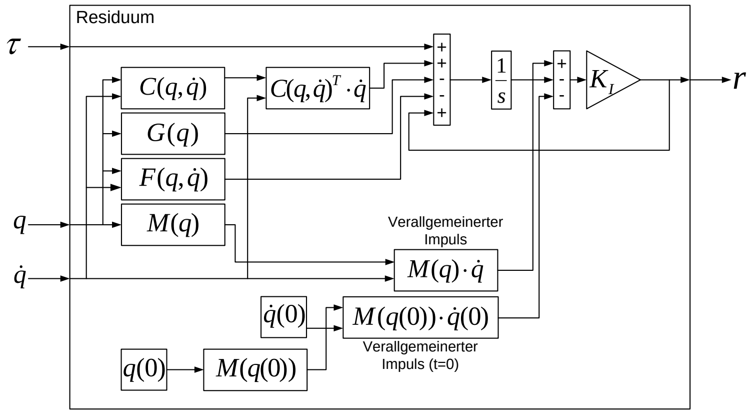


Bild 4.3.: Aufbau des in Matlab erstellten Residuums

Hier ist zu erkennen, dass aus den ermittelten Matrizen und den aktuellen Werten von $\boldsymbol{\tau}$, $\boldsymbol{q}$ und $\dot{\boldsymbol{q}}$ zunächst die Summe für das in Gleichung (3.23) zu erkennende Integral gebildet wird. Nachdem das Integral ermittelt ist, folgt das Addieren des verallgemeinerten Impulses zum aktuellen Zeitpunkt sowie das Subtrahieren des verallgemeinerten Impulses zum Zeitpunkt $t = 0$. Dieser so ermittelte Zwischenwert wird schließlich noch mit dem definierten Verstärkungsfaktor K_I multipliziert, um das endgültige Residuum zu erhalten.

Um mit dem Residuum die Kollisionsidentifizierung durchzuführen, muss der BioRob zunächst in eine Kollision geschickt werden, also eine Trajektorie abfahren können. Somit ist es zwingend notwendig, eine Regelung zu modellieren, die eine realitätsgetreue Bahntreue bietet.

4.1.2. Regler des elastischen 4-DoF BioRob mit Kollisionserkennung

Wie in Gleichung (3.6) zu sehen, wird jedes Gelenk durch zwei nichtlineare Differentialgleichung zweiter Ordnung beschrieben. Diese können auch durch ein System vier nichtlinearer Differentialgleichungen erster Ordnung dargestellt werden. Die Zustände dieses Systems führen zu einem Zustandsvektor der Form $[\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{\theta}, \dot{\boldsymbol{\theta}}]^T$. Der State-Space Regler, welcher nun genau diesen Zustand regelt, bedient sich der durch die Trajektorie vorgegebenen Werte für $\boldsymbol{q_d}$ und $\dot{\boldsymbol{q_d}}$, um mit den uns zur Verfügung stehenden Messwerten die Antriebsseite $(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$ sowie die Abtriebsseite $(\boldsymbol{q}, \dot{\boldsymbol{q}})$ zu Regeln. Dabei werden die Zustände beider Seiten (Antriebs- und Abtriebsseite) über einen P-Regler geregelt. Somit erhalten wir für jedes Gelenk folgendes Regelgesetz:

$$f_i = (q_{d,i} - q_i)kp_{q,i} + (\dot{q_{d,i}} - \dot{q_i})kp_{\dot{q},i} + (\theta_{d,i} - \theta_i)kp_{\theta,i} + (\dot{\theta_{d,i}} - \dot{\theta_i})kp_{\dot{\theta},i} \qquad (4.2)$$

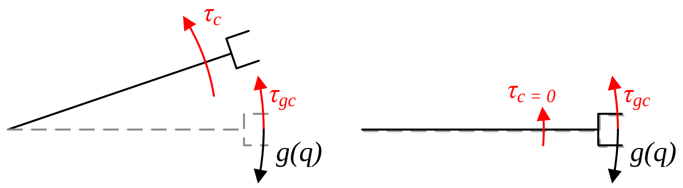


Bild 4.4.: Schematische Darstellung der Anteile der Gravitationskompensation (Moment τ_{gc}) und des Moments aus dem Zustandsregler (τ_c). Zusammen ergibt sich somit das vom Motor zu leistende Moment τ_m

Als Messwert steht der gesamte Zustand (also $[q_i, \dot{q}_i, \theta_i, \dot{\theta}_i]^T$) zur Verfügung. Leider gilt dies nicht für die Sollwertvorgaben der Trajektorie. Der Trajektorienplaner liefert nur Vorgaben für die Gelenkposition und -geschwindigkeit. Über die Motorposition und -geschwindigkeit stehen keine Informationen bereit. Es wird versucht, diese über folgende Überlegung anzunähern: Über die Vorgabe einer Gelenkposition $q_{d,i}$ soll der Regler diese statische Position anregeln. In dieser statischen Position wirken keine weiteren Effekte außer der Gravitation. Somit bestimmt die Gravitationskraft $g_i(\boldsymbol{q_d}) = \boldsymbol{g}(\boldsymbol{q_d}) \cdot \boldsymbol{e_i}$ wie stark die elastische Kopplung zwischen Gelenk und Glied gedehnt wird (z.B. bei horizontaler Lage), also wie stark bei stehendem Motor der Arm nach „unten" gezogen wird. Mit Hilfe dieser Überlegung erhalten wir folgenden Zusammenhang: $g_i(q_d) = k_{ei}(\theta_{d,i} - q_{d,i})$. Durch Umstellen der Formel zu $g_i(q_d)/k_{ei} + q_{d,i} = \theta_{d,i}$ kann $\theta_{d,i}$ ermittelt werden. Die Ableitung führt uns zum ebenfalls gesuchten $\dot{\theta}_{d,i}$. Nun ist es also möglich, das Regelgesetz zu verwenden.

Die Regelung im Gesamten betrachtet ist in zwei Teile aufgeteilt. Der eine Teil wird durch den eben beschriebenen State-Space Regler gebildet, der zweite Teil durch eine Gravitationskompensation (Bild 4.4). Die Gravitationskompensation hat die Aufgabe, die zur aktuellen Position wirkenden Gravitationskräfte heraus zu rechnen. Somit muss der Regler diese Kräfte nicht kennen und auch nicht berücksichtigen. Das hat den Vorteil, dass die Regelparameter unabhängig von der wirkenden Gravitation eingestellt werden können. Wollte man diese Kompensation mit im Regler verwirklichen, hat man das Problem, dass die wirkenden Kräfte je nach Konfiguration des Manipulators sehr unterschiedlich sind[1]. Um bei hohen Gravitationskräften schnell und genau zu reglen, müssten die Regelparameter höher gewählt werden. Dies hätte aber zur Folge, dass bei Konfigurationen, in denen kaum Gravitation wirkt, die Regelparameter keine schnelle und genaue Regelung zuließen.

Wie die einzelnen Teile der Regelung vom Trajektorienplaner bis zum Manipulator zusammen spielen, ist in folgender Übersicht (Bild 4.5) zu sehen.

[1] Beispielsweise wirken bei horizontaler Stellung des Armes hohe Gravitationskräfte. Zeigt der Arm jedoch nach unten, so sind diese fast Null.

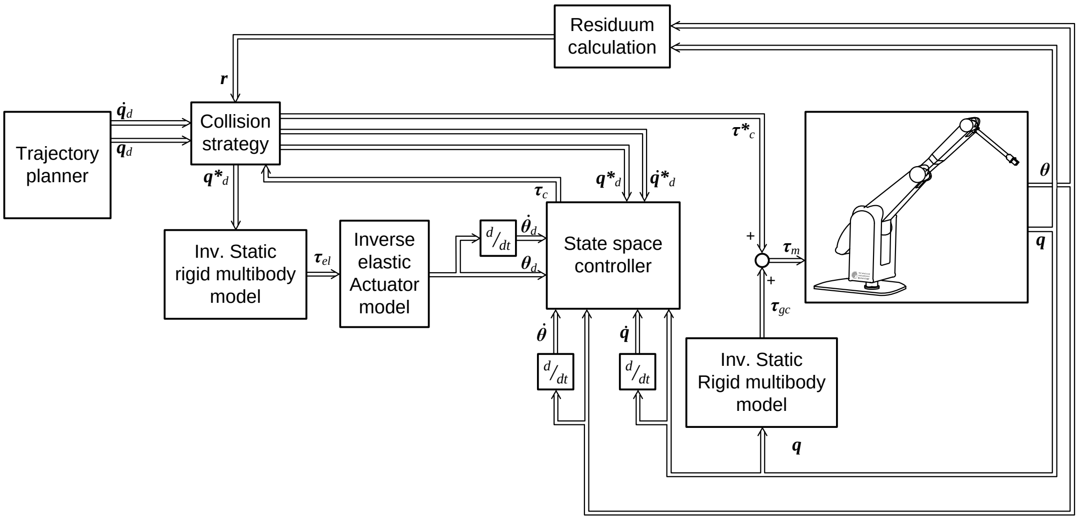


Bild 4.5.: Übersicht der Regelung des Manipulators

Zunächst gibt der Trajektorienplaner die Sollposition $\boldsymbol{q_d}$ und Sollgeschwindigkeit $\dot{\boldsymbol{q}}_{\boldsymbol{d}}$ der Gelenke vor. Diese werden anschließend in den Kollisionsstrategieblock geleitet. Der Kollisionsblock schleust die Sollwerttrajektorie $\boldsymbol{q_d}$ und $\dot{\boldsymbol{q}}_{\boldsymbol{d}}$ zum Regler unverändert durch, so lange keine Kollision vorliegt. Somit sind $\boldsymbol{q_d^*} = \boldsymbol{q_d}$, $\dot{\boldsymbol{q}}_{\boldsymbol{d}}^* = \dot{\boldsymbol{q}}_{\boldsymbol{d}}$. Aber auch das vom Zustandsregler geliefertes Moment bleibt unverändert ($\boldsymbol{\tau_c^*} = \boldsymbol{\tau_c}$), so lange keine Kollision vorliegt. Ist also keine Kollision detektiert, verhält sich der Kollisionsblock als würde er nicht existieren. Die für die Regelung benötigten Sollposition $\boldsymbol{\theta_d}$ und Sollgeschwindigkeit $\dot{\boldsymbol{\theta}}_{\boldsymbol{d}}$ des Motors, wird mit Hilfe des inversen Modells des elastischen Antriebs berechnet. Da die Regelung Gravitationskompensation voraussetzt, müssen die von der Gravitationskraft ausgelösten Gelenkmomente $\boldsymbol{\tau_{gc}}$ ermittelt werden. Dies geschieht über das inverse, statische Modell des starren Roboters und der aktuell gemessenen Gelenkposition $\boldsymbol{q}$. Zu diesem im Gelenk wirkenden Moment wird schließlich das vom Regler oder der Kollisionsstrategie vorgebene Gelenkmoment $\boldsymbol{\tau_c^*}$ addiert. Das Ergebnis bildet die endgültig auf die Motoren wirkenden Momente $\boldsymbol{\tau_m}$. Über eine Rückkopplung werden die gemessenen Gelenk- und Motorpositionen $\boldsymbol{q}$ und $\boldsymbol{\theta}$ und deren Ableitungen $\dot{\boldsymbol{q}}$ und $\dot{\boldsymbol{\theta}}$ in den Regelkreislauf geleitet.

Wird nun aber der Betrag des Residuums $\boldsymbol{r}$ größer als ein gewählter Schwellwert, verändert der Kollisionsblock passend zur jeweiligen Strategie die Werte der Ausgänge.

- **Strategie 0:**

$$\boldsymbol{q_d^*} = \boldsymbol{q_d}, \quad \dot{\boldsymbol{q}}_{\boldsymbol{d}}^* = \dot{\boldsymbol{q}}_{\boldsymbol{d}}, \quad \boldsymbol{\tau_c^*} = \boldsymbol{\tau_c}$$

Da bei dieser Strategie keine Reaktion gezeigt werden soll, werden wie im Normalbetrieb die Sollwerte und das Moment aus dem Zustandsregler einfach weiter geleitet, und die Trajektorie fortgeführt.

- **Strategie 1:**

$$\boldsymbol{q_d^*} = \boldsymbol{q_{kol}} = const, \quad \dot{\boldsymbol{q_d^*}} = \dot{\boldsymbol{q_{kol}}} = const, \quad \boldsymbol{\tau_c^*} = \boldsymbol{\tau_c}$$

  Um die Strategie des sofortigen Stoppens (Anhalten der Trajektorie) des Manipulators zu realisieren, werden nach dem Detektieren einer Kollision die zu diesem Zeitpunkt gemessene Gelenkposition sowie deren Ableitung als Gelenkgeschwindigkeit dem Regler als Sollvorgabe übergeben. Somit ist ein schnellstmögliches Stoppen ohne die Verwendung von mechanischen Arretierungen, welche nicht vorhanden sind, möglich. Das für diese Vorgabe vom Zustandsregler berechnete Motormoment wird ohne Veränderung zum Regelkreislauf weitergeleitet.

- **Strategie 2:**

$$\boldsymbol{\tau_c^*} = \boldsymbol{0}$$

  In Strategie 2 wird die Positionsregelung deaktiviert, indem der Anteil des zustandsreglers entfernt wird. Somit wirkt nach der Kollision nur noch die Gravitationskompensation. Um das möglichst einfach im Strategieblock zu implementieren, wird als vom Regler angesteuertes Moment einfach Null ausgegeben. Da hierdurch der Zustandsregleranteil keinen Einfluss auf den Regelkreislauf mehr hat, können die Sollwertvorgaben ohne Veränderung durch den Block geschleust werden.

- **Strategie 3:**

$$\boldsymbol{\tau_c^*} = \boldsymbol{K_R r}$$

  Bei dieser Strategie geht es darum, den Arm aus der Kollision heraus zu bewegen, indem das Residuum in die Momentenregelung mit eingebunden wird. Das Residuum spiegelt die Richtung der Kollisionskraft in Gelenkkoordinaten wieder. Um nun aus der Kollision zu fahren, muss das Residuum, als Vorgabe für die entgegen gesetzte Kollisionskraft in den Motoren genutzt werden. Der Parameter K_R dient dazu, diese Kraft so zu skalieren, dass das vom Residuum vorgegebene Moment weder zu groß, noch zu klein ist. Auch hier können die Sollvorgaben wieder unverändert an den Regler weiter gegeben werden, da die ermittelten Werte des Zustandsreglers nach Auftreten der Kollision nicht mehr verwendet werden.

- **Strategie 4:**

$$\boldsymbol{q_d^*} = \boldsymbol{q_{kol}} + \int \boldsymbol{K_R r}, \quad \dot{\boldsymbol{q_d^*}} = \boldsymbol{K_R r}, \quad \boldsymbol{\tau_c^*} = \boldsymbol{\tau_c}$$

  In der letzten betrachteten Strategie wird ähnlich wie bei der vorigen vorgegangen. Auch hier soll der Roboterarm aus der Kollision gefahren werden. Ein anschließendes Auslaufen soll jedoch verhindert werden. Das Residuum wird also nun nicht als Momentvorgabe für die Motoren, sondern als Sollwertvorgabe für die Gelenkgeschwindigkeit verwendet. Somit wird der Manipulator, so lange das Residuum nicht Null ist, aus der Kollision beschleunigt. Auch hier dient der Parameter $\boldsymbol{K_R}$ wieder dazu, die Werte von $\boldsymbol{r}$ an dessen Aufgabe anzupassen. Um auch die zu dieser Geschwindigkeit passenden Gelenkposition zu erhalten, muss das Integral von $\boldsymbol{K_R r}$ gebildet werden. Da dies jedoch erst ab dem Zeitpunkt der Kollision geschieht, muss der Wert des Integrals auf die Position zum Zeitpunkt der Kollision addiert werden. Das mit diesen Vorgaben ermittelte Moment des Zustandsreglers kann dann unverändert zum Regelkreislauf geführt werden.

4.2. Simulation

Die Simulation hat den Zweck, das konstruierte Modell zu beurteilen, ohne es direkt auf dem Prototyp ausführen zu müssen. Somit ist gewährleistet, dass eventuelle Fehlannahmen oder Ungenauigkeiten im Modell, den Prototypen nicht beschädigen. Zudem bietet Matlab/Simulink die Möglichkeit, Änderungen schnell einzupflegen und und deren Auswirkungen zu betrachten. In unserem Fall benötigen wir die Simulation, um herauszufinden, ob die angewendete Kollisionsdetektion wie gewünscht funktioniert, aber vor allem, um die Auswirkungen einer Kollision zu betrachten. Dies geschieht nur im Rahmen des leicht vereinfachten Modells, welches für die Simulation verwendet wird. Da das Modell zwar eine Vereinfachung darstellt, jedoch trotzdem fast alle Einflüsse der Realität berücksichtigt[2], ist es möglich, Schlüsse vom Modell auf die Realität zu übertragen.

Besonderes Augenmerk der Simulation liegt auf dem Verhalten des Manipulators mit und ohne eine Kollisionsstrategie. Dabei werden die auftretenden Gelenkmomente sowie die Gelenkgeschwindigkeiten in Bezug auf die verschiedenen Strategien betrachtet. Auch die Reaktionsgeschwindigkeit soll abgeschätzt werden[3]. Um beurteilen zu können, welche Gefahr für den Menschen, aber auch für die Motoren, vom Manipulator ausgeht, sollen schließlich noch die auftretenden externen Kräfte inspiziert werden.

4.2.1. Experiment

Mit einem einfachen Experiment soll nun die Reaktion des Manipulators auf eine Kollision beurteilt werden. Dabei fährt der BioRob eine Kreisbahn mit Gelenk 2 (Schultergelenk) mit $10°/s$ und $100°/s$ (entspricht einer linearen Endeffektorgeschwindigkeit von ca. $0.14\frac{m}{s}$ sowie $1.44\frac{m}{s}$) ab. In dieser Kreisbahn liegt ein Hindernis, sodass der Endeffektor mit diesem kollidiert. Das Hindernis hat eine Elastizität von 1000 N/m sowie eine Dämpfung von $10\frac{N}{m/s}$. Bei einer Eindringtiefe von $1cm$ wird eine Kraft von $1000\frac{N}{m} \cdot 0.01m = 10N$ verursacht. Bei einer Eindringgeschwindigkeit von $1\frac{m}{s}$ wird zusätzlich noch die Kraft $10\frac{N}{m/s} \cdot 1\frac{m}{s} = 10N$ verursacht. Solch eine Elastizität und

[2]Beispielsweise wird der Effekt der Haftreibung in den Gelenken und Motoren nicht modelliert. Dies hat jedoch keine negativen Auswirkungen, da diese Reibungen das Schwingen des Manipulators kurz vor dem Stillstand sogar dämpfen würden. Demnach dauert das simulierte Schwingverhalten kurz vor einer Ruheposition länger an als es in Wirklichkeit auftreten wird.

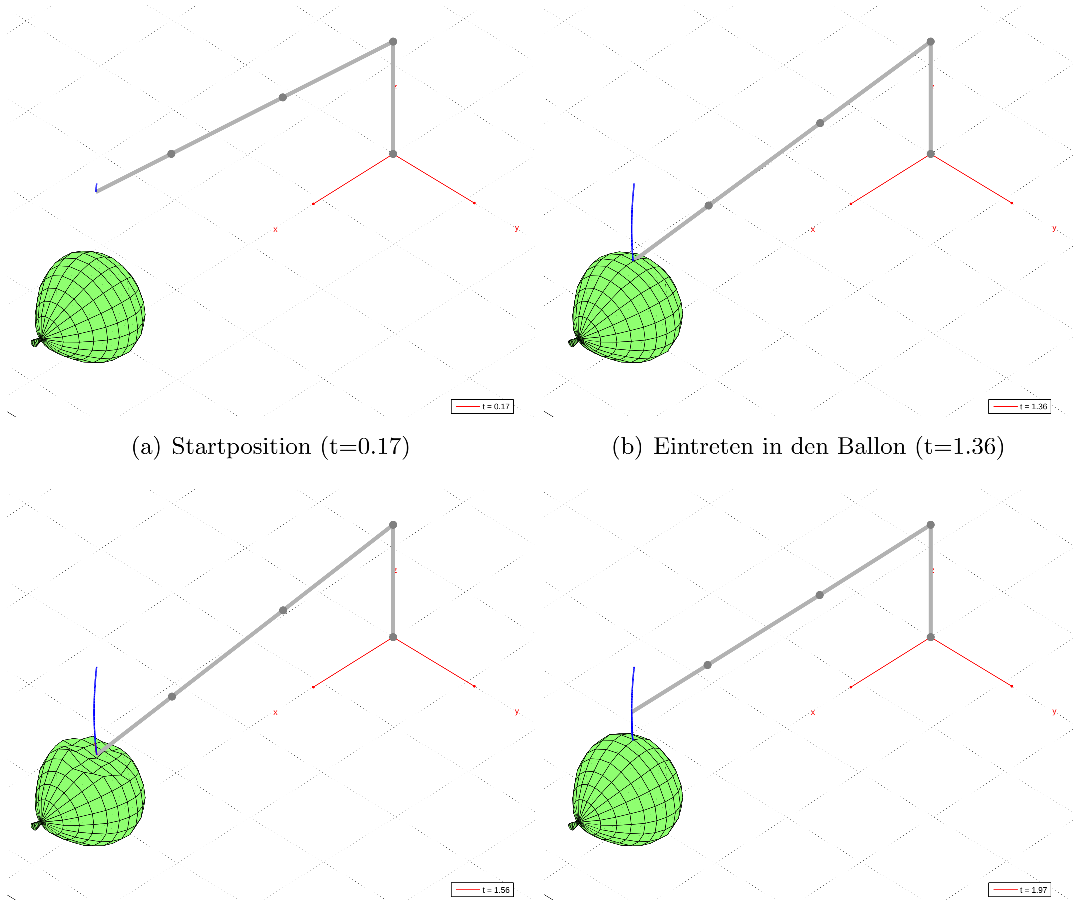
[3]dies ist leider nur bedingt möglich, da die Verzögerungen über Motoransteuerung und Motorreaktionszeit nicht im Modell berücksichtigt werden

Dämpfung kann man sich an einer Art Ballon oder Ball vorstellen. Aus ästhetischen Gründen ist daher das Hindernis als Ballon dargestellt.

Die Verstärkungsmatrix bzw. die Matrix der Zeitkonstanten des Residuums wurde mit $\boldsymbol{K_I} = 150 \cdot \boldsymbol{I}$ gewählt. Dieser Wert hatte zwar schon numerisches Rauschen zur Folge, dieses lag jedoch in einer Größenordnung von $1 \cdot 10^{-9}$, welches keinen Einfluss auf das Verhalten des Manipulators hatte. Die beiden Verstärkungsmatrizen für die Reaktionsstrategien drei und vier wurden mit $\boldsymbol{K_R} = 0.08 \cdot \boldsymbol{I}$ für Strategie 3 und mit $\boldsymbol{K_R} = 0.03 \cdot \boldsymbol{I}$ für Strategie 4 gewählt. Bei der Wahl des Parameters für Strategie 3 musste darauf geachtet werden, dass die auf die Motoren gegebene Kraft zum Ausfahren aus der Kollision, nicht zu groß ist. Bei der Kollision mit $100°/s$ hatte dies zur Folge, dass der Manipulator eine noch zu hohe Geschwindigkeit besaß, mit der er in die Gravitationskompensation lief. Dies wiederum hatte zur Folge, dass der Manipulator sich mehr als 180° um Gelenk zwei drehte, bevor er zum Stillstand gekommen ist. Ein solches Ausweichen könnte weitere unvorhergesehene Kollisionen nach sie ziehen. Daher wurde $\boldsymbol{K_R}$ so gewählt, dass der Manipulator maximal um 90° aus der Kollision zurück gefahren wird. Somit ergibt für $\boldsymbol{K_R}$ ein Obergrenze. Für langsame Geschwindigkeiten wie z.B. $10°/s$ hätte ein größerer Faktor ein schnelleres Austreten aus der Kollision zur Folge, da aber ein Parameter für alle Geschwindigkeiten gesucht wurde, musste man sich an den schnellen Bewegungen orientieren. Bei Strategie 4 sollte der Verstärkungsparameter den Arm so schnell wie möglich aus der Kollision beschleunigen, den Arm aber durch eine zu hohe Beschleunigung nicht unnötig lange in Bewegung halten. Da die Strategie nur bei hohen wirkenden Kräften, also auch nur bei hohen Geschwindigkeiten, große Beschleunigungen entstehen lässt, wurde der Parameter ebenfalls bei einer schnellen Bewegung eingestellt. Dabei wurde er so weit verkleinert, dass eine weitere Vergrößerung keine merkliche Verringerung der Zeit bis zum Austritt aus der Kollision zur Folge hatte. Für langsame Geschwindigkeiten, und somit auch kleinen auftretenden Kräften, hatte diese Feineinstellung keine Auswirkungen. Selbst eine Vergrößerung, des nun gewählten Parameters, um den Faktor 100 hatte nur eine geringe Veränderung des Verhaltens zur Folge.

Als Schwelle für das Erkennen einer Kollision wurde festgelegt, dass die Norm des Residuumsvektors 1 % des maximalen Ausgangsmoments der Motoren übersteigen muss. Allgemein muss ein Schwellwert bestimmt werden; denn durch Modellierungsfehler, nicht modellierte Effekte und Fehler durch numerische Berechnungsschritte, können im Residuum Momente auftreten, die jedoch nicht von einer Kollision herrühren. In Bild 4.6 ist der Ablauf des geschilderten Versuchs zu sehen. Hierbei wird Strategie 4 als Reaktion auf die Kollision verwendet. Der Manipulator hat hier eine Geschwindigkeit von $10°/s$, welche nur am zweiten Gelenk anliegt. Die Bahn des Endeffektors ist in blau eingezeichnet.

Zur Veranschaulichung des Verfahrens sind in Bild 4.7 die Komponenten des Residuumvektors zu sehen. Nachdem die Norm des Residuumvektors 1 % (0.1 Nm) des maximalen Ausgangsmoments der Motoren überschreitet, wird die Kollision erkannt. Dies veranlasst den Manipulator aus dem Ballon zu fahren, wodurch das Residuum wieder auf Null zurück sinkt.



(a) Startposition (t=0.17)

(b) Eintreten in den Ballon (t=1.36)

(c) Eintrittsposition (t=1.56)

(d) Manipulator ausgetreten (t=1.97)

Bild 4.6.: Kollision des BioRob mit einer Art Ballon (bei $10°/s$) und anschließender Reaktion (Strategie 4) sobald die Norm des Residuumsvektors 1% des maximalen Ausgangsmoments der Motoren (0.1Nm) übersteigt

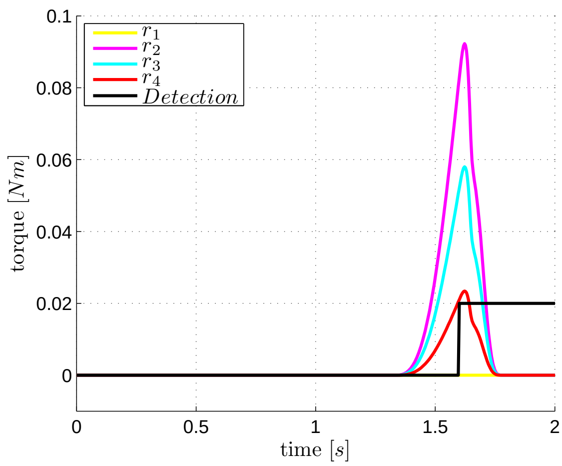


Bild 4.7.: Komponenten des Residuumvektors inkl. Zeitpunkt der Kollisionserkennung

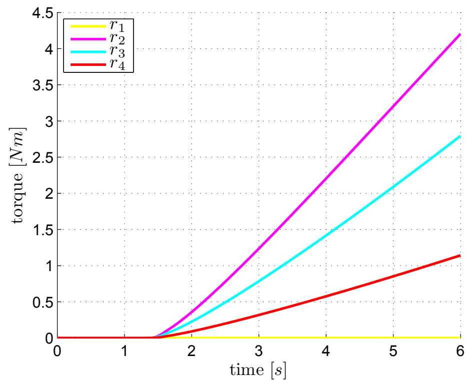
In diesem Bild zeigt sich zudem auch, dass die Momente im Gelenk zwei bei diesem Experiment am größten werden. Das liegt daran, dass das Moment aus der Kollision der kinematischen Kette entlang bis zum Gelenk zwei weiter geleitet wird. Da dort das Glied senkrecht anschließt, kann das Moment nicht mehr im Gelenk eins, welches um die horizontale Achse dreht, übertragen werden. Aus diesem Grund bleibt die erste Komponente des Residuumvektors (r_1) kontinuierlich auf Null.

Im folgenden Abschnitt werden die Ergebnisse der auf der geschilderten Simulation erfassten Daten diskutiert. Um verschiedene Arten der Kollision zu betrachten wurden die Parameter des Hindernisses verändert. So war es möglich, auch Kollisionen mit harten Gegenständen, wie z.B. dem menschlichen Schädel, anzunähern. Dies wiederum war die Grundlage, um den Sicherheitsaspekt genauer zu betrachten.

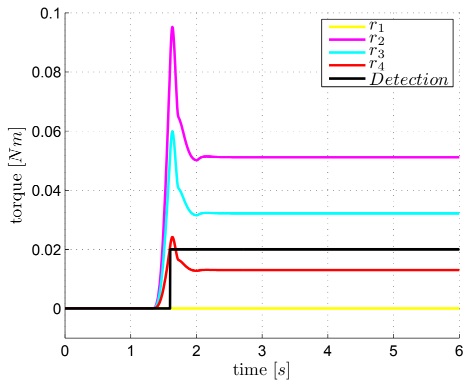
4.2.2. Auswertung

Um einen Eindruck davon zu bekommen, wie sich die Kollisionskraft am Endeffektor auf die einzelnen Gelenke aufteilt, betrachten wir zunächst die einzelnen Elemente des Residuumvektors, welche die Momente in den einzelnen Gelenken widerspiegeln. Die Kollision geschieht mit dem im vorigen Abschnitt beschriebenen weichen Hindernis (z.B. einem Luftballon), mit einer langsamen Geschwindigkeit von $10°/s$.

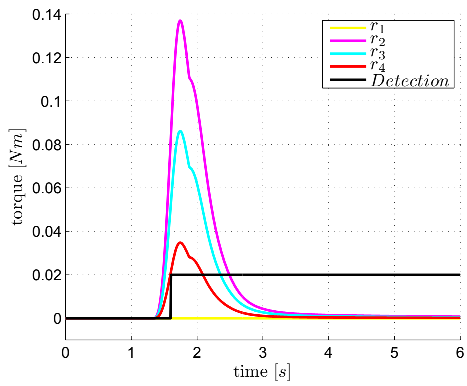
In den Abbildungen im Bild 4.8 ist zu erkennen, dass das Moment im Gelenk zwei am größten wird. Dies gilt auch für Kollisionen mit schnelleren Manipulatorgeschwindigkeiten. Aus diesem Grund werden im Folgenden nur noch die Momente im zweiten Gelenk betrachtet. Auch die vermuteten Verhalten der verschiedenen Strategien sind in den Abbildungen wieder zu erkennen.



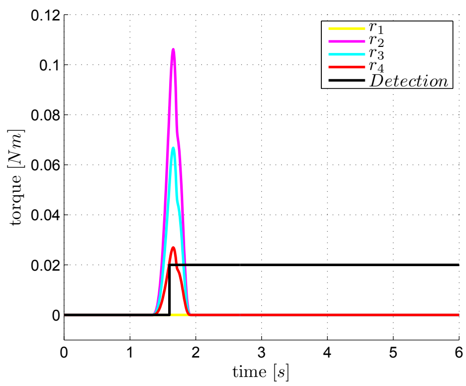
(a) Komponenten des Residuumvektors bei einer Kollision mit Strategie 0



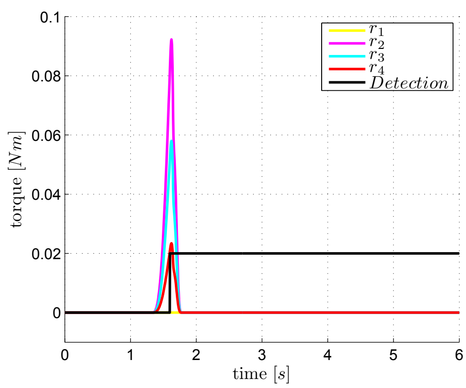
(b) Komponenten des Residuumvektors bei einer Kollision mit Strategie 1, inkl. des Zeitpunkts der Kollisionserkennung



(c) Komponenten des Residuumvektors bei einer Kollision mit Strategie 2, inkl. des Zeitpunkts der Kollisionserkennung



(d) Komponenten des Residuumvektors bei einer Kollision mit Strategie 3, inkl. des Zeitpunkts der Kollisionserkennung

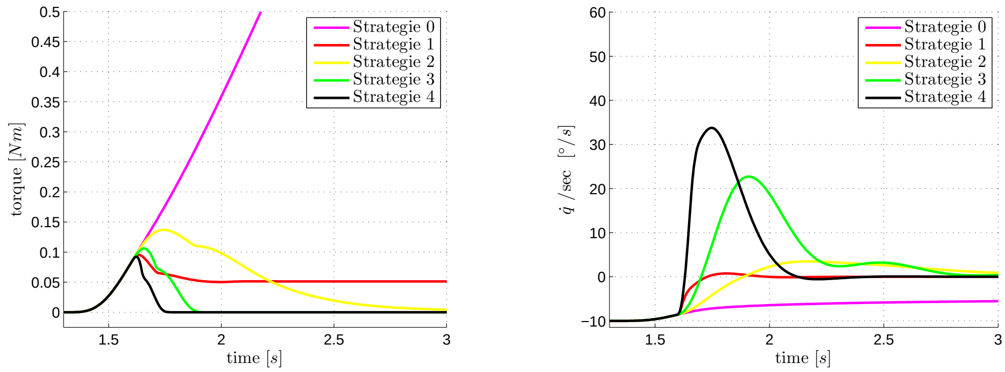


(e) Komponenten des Residuumvektors bei einer Kollision mit Strategie 4, inkl. des Zeitpunkts der Kollisionserkennung

Bild 4.8.: Komponenten des Residuumvektors bei der Kollision mit einem „Luftballon“ mit $10°/s$

Die Gelenkmomente bei einer Kollision mit Strategie 0 (Bild 4.8 (a)) steigen immer weiter an bis das maximale Motormoment erreicht ist. Da nicht auf die Kollision reagiert wird, versucht der Regler, die immer größer werdende Abweichung zur Sollposition des Endeffektors, mit einem ebenfalls immer größer werdenden Motormoments auszugleichen. Da bei einer so weichen Kollision diese erst erkannt wird, wenn der Endeffektor bereits in das Hindernis eingedrungen ist, führt das Stoppen des Manipulators (Strategie 1 / Bild 4.8 (b)) zu einer kontinuierlichen Kraft in das Hindernis. Aus diesem Grund lassen hier die Gelenkmomente nie nach. In Bild 4.8 (c), (d) und (e) sind die reaktiven Strategien 2 bis 4 zu sehen. Da diese den Manipulator aus der Kollision fahren, gehen analog dazu auch die Gelenkmomente zurück auf Null. Zu erahnen ist hier schon, dass Strategie 4 die schnellste ist, da bei ihr die Momente zum frühsten Zeitpunkt wieder Null erreicht haben.

Wie schon erwähnt, wurde das eben beschriebene Experiment in den zwei Geschwindigkeiten $10°/s$ und $100°/s$ mit jeder der vorgestellten Reaktionsstrategien durchgeführt. Es folgt der direkte Vergleich der verschiedenen Strategien bei Verwendung dieser Geschwindigkeiten.

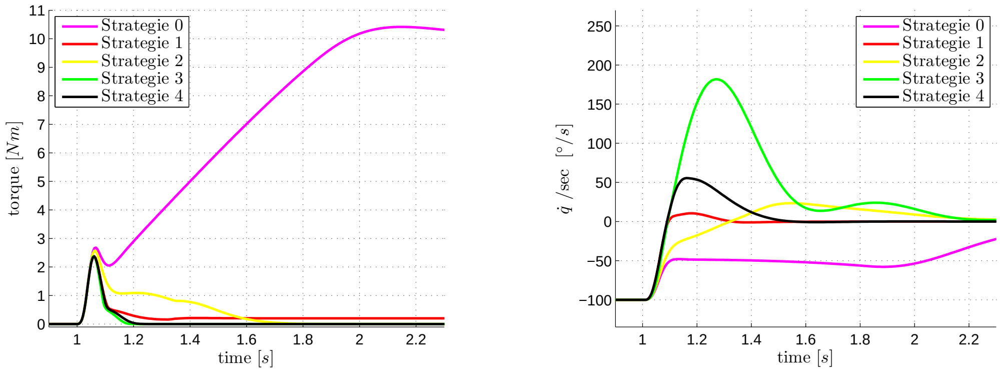


(a) Residuum r_2 am zweiten Gelenk mit den verschiedenen Kollisionsstrategien

(b) Geschwindigkeit $\dot{q}_2$ am zweiten Gelenk mit den verschiedenen Kollisionsstrategien

Bild 4.9.: Residuum und Geschwindigkeit im zweiten Gelenk bei $10°/s$

In Bild 4.9 sind die Kollisionen bei langsamer Geschwindigkeit zu sehen. Dabei sind in Abbildung (a) das berechnete Residuum des zweiten Gelenks sowie in Abbildung b) die Geschwindigkeiten dieses Gelenks zu sehen. Zur einfacheren Gegenüberstellung der einzelnen Strategien sind diese jeweils alle in (a) und (b) geplottet. Für die durch die einzelnen Strategien erreichten Gelenkmomente ist aus Abbildung (a), wie bereits erwähnt, zu erkennen, dass das Moment immer weiter ansteigt. Hier nicht zu sehen, da es außerhalb des Bildbereichs liegt, ist, dass nach dem Erreichen des maximalen Motormoments, dieses gehalten wird. Auch für Strategie 1 erkennt man wieder, dass das Verweilen in der Kollision ein kontinuierliches Moment in dem Gelenk verursacht. Für alle reaktiven Strategien (Strategien 2 bis 4) geht das Moment auf Null zurück, weil der Manipulator aus der Kollision gefahren wird. Dabei ist Strategie 4 die schnellste. Das wiederum ist nicht selbstverständlich. Durch die Wahl eines großen Verstärkungsparameters $\boldsymbol{K_R}$ für Strategie 3 kann das Verhalten dieser Strategie an Strategie 4 angepasst werden. Jedoch verlängert sich der Zeitpunkt bis zum Stillstand des Manipulators dementsprechend, da eine höhere Geschwindigkeit beim Zeitpunkt des Schaltens auf die Gravitationskompensation ein längeres Auslaufen nach sich zieht. Die Zeitpunkte bis der Manipulator zum Stillstand gekommen ist, können aus Abbildung (b) gelesen werden. Vor der Kollision beträgt die Geschwindigkeit die vorgegebenen $10°/s$. Diese wird nach Erkennen der Kollision von Strategie 1 bis 4 auf Null reduziert. Da Strategie 0 immer weiter Kraft in die Kollision gibt bis die Motoren in Sättigung geraten, dreht sich auch der Manipulator langsam weiter und erreicht in diesem Zeitraum nicht Null. Strategie 3 schafft es durch die lange Auslaufzeit nur knapp vor Strategie 2 den BioRob zum Stillstand zu bringen.



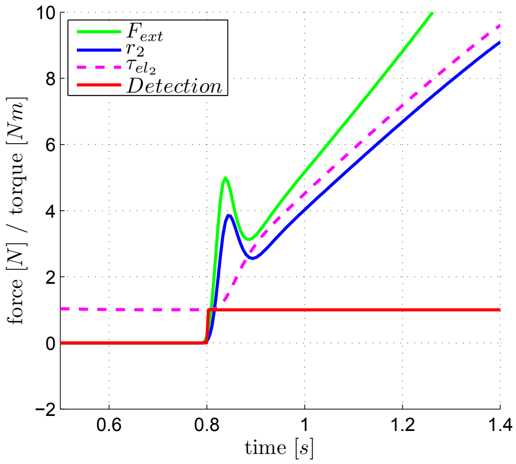
(a) Residuum r_2 am zweiten Gelenk mit den verschiedenen Kollisionsstrategien

(b) Geschwindigkeit $\dot{q}_2$ am zweiten Gelenk mit den verschiedenen Kollisionsstrategien

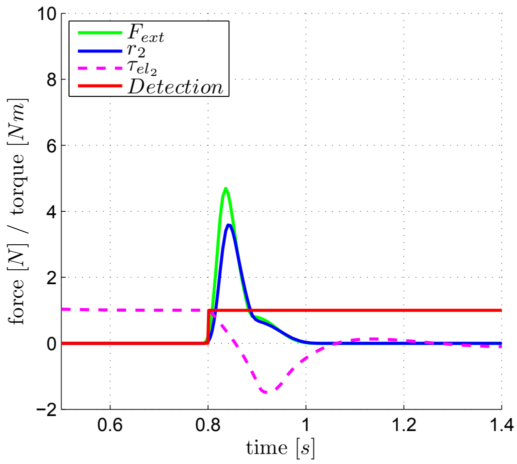
Bild 4.10.: Residuum und Geschwindigkeit im zweiten Gelenk bei $100°/s$

Für die Kollisionen bei schneller Geschwindigkeit (Bild 4.10) bietet sich ein ähnliches Bild. Jedoch schafft es Strategie 3, das wirkende Gelenkmoment etwas schneller als Strategie 4 zu eliminieren. Aber auch hier gilt, dass Strategie 3 erst fast zeitgleich mit Strategie 2 den Manipulator zum Stehen bringt. Somit ist Strategie 3 für hohe Geschwindigkeiten zwar etwas schneller als Strategie 4, aber dafür benötigt Sie fast doppelt so lange, um die Geschwindigkeit des Manipulators auf Null zu reduzieren. Strategien 0 und 1 verhalten sich wie im vorherigen Fall. Im Allgemeinen gelingt es Strategien 1 und 4 am schnellsten, den BioRob zum Stehen zu bringen, jedoch liegt bei Strategie 1 weiterhin eine Kollision vor.

Wie in Kapitel 2 schon beschrieben, wirken die Trägheitsmoment von den Motoren/Getrieben erst verzögert über die elastische Kopplung in die Kollision. Die Auswirkungen dieser Verzögerung ist in Bild 4.11 Abbildungen (a) und (b) zu erkennen. Zur Orientierung sind die Kraft am Endeffektor F_{ext} sowie das Residuum des zweiten Gelenks r_2 aufgetragen. Das Moment, welches von der elastischen Kopplung auf das Glied wirkt, ist durch die gestrichelte Linie dargestellt. In Abbildung (a) wird auf die Kollision nicht reagiert (Strategie 0). Zu erkennen ist die zum Zeitpunkt des Aufpralls wirkende Kraft, welche sich auch im Residuum bemerkbar macht. Nachdem diese abgegeben ist, wirkt die Kraft der elastischen Kopplung in die Kollision, da die Motoren immer weiter drehen. Vergleichen wir nun (a) mit (b) stellt man fest, dass nach der Reaktion auf die Kollision (Strategie 4) die Motoren sofort gegen die Kollisionsbewegung arbeiten und somit die Kraft der elastischen Kopplung in die andere Richtung wirken lassen. Dadurch wird das verzögerte Einwirken des Moments in die Kollision verhindert, welches für dieses Experiment eine größere externe Kraft hervorrufen würde als die Kollision selbst. Leider ist es aber durch dieses sofortige Gegensteuern der Motoren nicht möglich, die Kollisionskraft vollständig auszulöschen. Eine Linderung dieser ist aber zu erkennen.



(a) Residuum r_2 am zweiten Gelenk, Moment τ_{el_2} der elastischen Kopplung und von extern wirkende Kraft F_{ext} bei Strategie 0

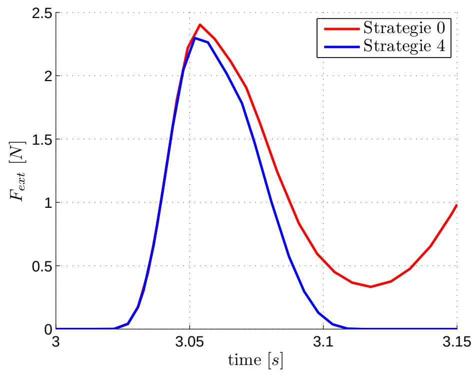


(b) Residuum r_2 am zweiten Gelenk, Moment τ_{el_2} der elastischen Kopplung und von extern wirkende Kraft F_{ext} bei Strategie 4

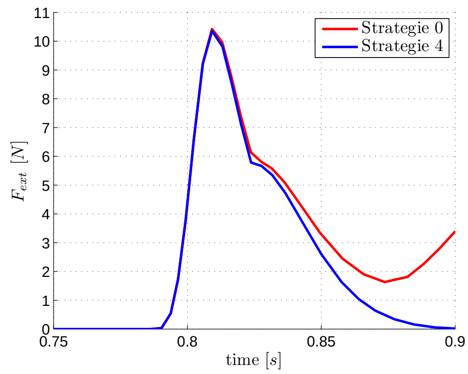
Bild 4.11.: Reduktion von Motor/Getriebe-Momenten durch eine Kollisionsreaktionsstrategie. Kollision mit dem „Ballon“ mit ca. 2m/s

Wie eben gesehen, ist es über eine geeignete Strategie möglich zu verhindern, dass die Kräfte der elastischen Kopplung mit in die Kollision wirken. Aber nicht nur das ist möglich. Auch die während dem ersten Kontakt übertragene Kollisionskraft kann in einem gewissen Rahmen abgeschwächt werden. Durch die hohe Elastizität hat die Strategie genügend Zeit, über die Motoren auf die Kollision zu reagieren.

In Bild 4.12 (a) und (b) ist zu sehen, in welchem Rahmen es möglich ist, die Kollisionskraft zu verringern. Abbildung (a) zeigt den schon bekannten Versuchsaufbau aus Bild 4.6, jedoch ist das Hindernis 1000mal steifer. Bei solch harten Gegenständen geschieht die Übertragung der Kollisionskraft in so kurzer Zeit, dass es nur bei relativ geringen Kollisionsgeschwindigkeiten möglich ist, schnell genug auf die Kollisionen zu reagieren. So ist bei einer Geschwindigkeit von 0.5m/s schon nur noch schwer eine Verringerung in Abbildung (a) zu erkennen. Es gilt aber, dass Kollisionen mit maximal dieser Geschwindigkeit (sogar bis ca. 0.7m/s) erfolgreich abgeschwächt werden können. Bei Kollisionen mit weniger steifen Hindernissen kann dann die Geschwindigkeit wiederum höher ausfallen. In Abbildung (b) ist zu sehen, dass eine Verringerung der



(a) Kollision mit einem harten Gegenstand (Steifheit von $10^6 N/m$) mit einer langsamen Geschwindigkeit ($32°/s$ ca. $0.5m/s$)



(b) Kollision mit einem etwas weicheren Gegenstand (Steifheit von $10^5 N/m$) mit einer schnellen Geschwindigkeit ($128°/s$ ca. $2.0m/s$)

Bild 4.12.: Reduktion der Kollisionskraft durch Strategie 4

Steifheit um den Faktor 10 schon genügt, sodass die Kollisionskraft bei einer Kollision mit bis zu 2m/s reduziert werden kann. Diese Möglichkeit der Reduktion schützt aber nicht nur die Person, welche in die Kollision verwickelt ist. Aus der geringeren Kollisionskraft folgt auch, dass die Kraft in den Gelenken des Manipulators weniger stark ausfällt und somit die Gelenke und Getriebe geschont werden.

Doch welche Gefahr geht allgemein von den bei der Kollision erzeugten Kräften für die Menschen aus? Fortan betrachten wir die Kollisionskräfte in Bezug darauf, ob es möglich ist, durch diese Frakturen am Kopf hervorzurufen. Zur Orientierung betrachten wir zwei Knochen des menschlichen Schädels.

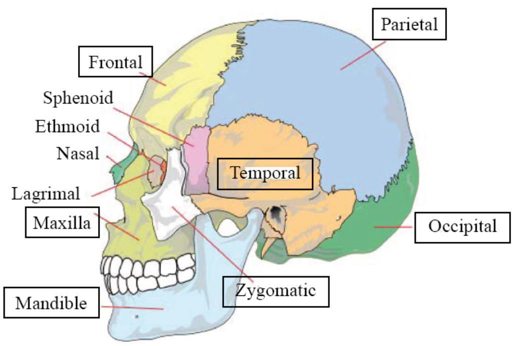
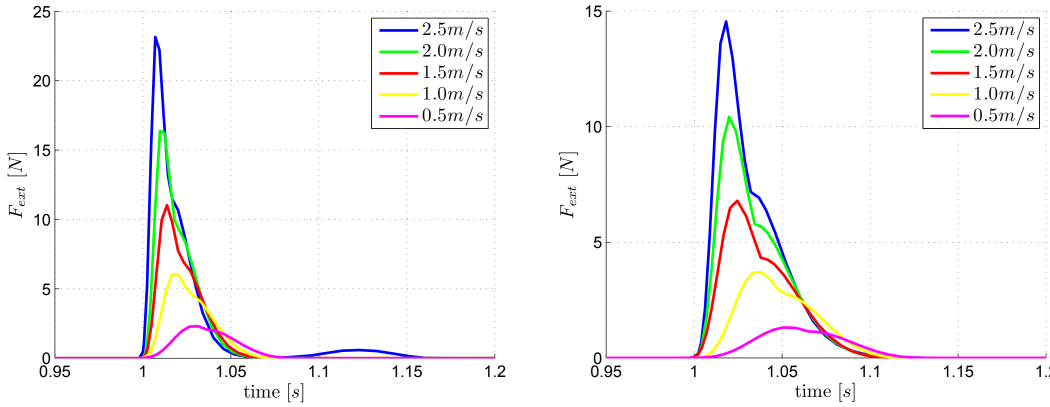


Bild 4.13.: Vereinfachte Anatomie des menschlichen Kopfes (aus [6])

Knochen	Knochenbruch ab
Mandible (A-P)	1.78kN
Mandible (lateral)	0.89kN
Maxilla	**0.66kN**
Zygoma	0.89kN
Frontal	**4.0kN**
Temporo-Parietal	3.12kN
Occiput	6.41kN

Tabelle 4.2.: Toleranzen des menschlichen Schädels (aus [6])

Das Stirnbein („Frontal“) ist der stabilste Gesichtsknochen. Wie aus Tabelle 4.2 zu erkennen, benötigt man eine Kraft von 4000 N, um eine Fraktur hervorzurufen. Im Gegensatz zum Stirnbein ist der Oberkieferknochen („Maxilla“) der am wenigsten stabilste Knochen. Dieser bricht schon ab einer Kraft von 660N. Die Frage ist nun, ob es bei üblichen Geschwindigkeiten des Endeffektors überhaupt möglich ist, eine Fraktur zu bewirken. Um dies zu ermitteln, stellten wir im Hindernis aus dem bekannten Versuchsaufbau annähernd dieselbe Steifigkeit wie die der Knochen ein. Da in der Regel die größte Kraft in den ersten Millisekunden der Kollision übertragen wird, und zu diesem Zeitpunkt das Hindernis noch nicht bewegt wurde, ist unser Versuchsaufbau eine hinreichend genaue Annäherung an die in Realität übertragene Kraft bei solch einer Kollision. Demnach wurden die Hindernisse so eingestellt, dass sie die Steifheit des Stirnbeins ($10^6 N/m$) sowie des Oberkiefers ($10^5 N/m$) hatten.



(a) Kollision zwischen BioRob und Stirnbein (Steifheit von $10^6 N/m$)

(b) Kollision zwischen BioRob und Oberkiefer (Steifheit von $10^5 N/m$)

Bild 4.14.: Kollision des BioRob mit dem Stirnbein und Oberkiefer. Die betrachteten Kollisionsgeschwindigkeiten befinden sich im Intervall von 0.5m/s bis 2.5m/s in jeweils 0.5m/s Schritten.

Die Messungen dieser Experimente sind in Bild 4.14 zu sehen. Wie schon in vorherigen Experimenten festgestellt, erkennt man, dass mit steigender Einschlagsgeschwindigkeit sich die Zeit der Kraftübertragung verringert, sowie die Kraft selbst zunimmt. Im Vergleich von Abbildung (a) und (b) kann man ausmachen, dass die maximal auftretende Kraft jeweils bei einer Kollision von 2.5m/s liegt. Des Weiteren ist die Kollisionskraft beim etwas elastischeren Oberkiefer für alle Geschwindigkeiten geringer als die Kollisionskräfte am Stirnbein. Die wichtigste Erkenntnis dieses Experiments ist jedoch, dass die höchste Kollisionskraft bei ca. 23 N (Abbildung (a)) liegt. Diese Kraft befindet sich in einem Bereich weit unter den in Tabelle 4.2 angegebenen Grenzen für eine Fraktur (mind. 660N) des Stirnbeins oder des Oberkiefers. Demnach besteht nach dieser Annäherung keinerlei Gefahr für Frakturen, bei stumpfen Kollisionen. Hier sei erwähnt, dass dies für Kollisionen, bei denen der Kopf z.B. zwischen Manipulator und einer Wand eingequetscht ist, nicht gilt. Bei solchen Kollisionen zählt nicht nur die übertragende Kraft, sondern auch die Beschleunigung in die Klemmung sowie der Weg des Manipulators, bis keine Bewegung in die Kollision mehr ausgeführt wird. Außerdem können Verletzungen, welche durch scharfe Gegenstände hervorgerufen werden, ebenfalls sehr bedrohlich werden.

4.3. Simulationsergebnisse

Wir haben gezeigt, dass es generell mit diesem Verfahren möglich ist, eine Kollision frühzeitig zu erkennen und auf diese angemessen zu reagieren. Aus den fünf hierfür vorgestellten Strategien kristallisierten sich die reaktiven Strategien (2, 3 und 4) als brauchbar heraus. Nach der Erkennung einer Kollision wirken sie keine weitere Kraft in die Kollision ein bzw. fahren den Manipulator automatisch aus der Kollision. Am effektivsten erschien hier Strategie 4, denn sie brachte den BioRob nach dem Herausfahren am schnellsten zum Stehen und verringerte ebenfalls sehr schnell das in den Gelenken wirkende Kollisionsmoment.

Wie wir aus den besprochenen Simulationen erkennen können, haben sich einige Vermutungen aus dem Vorfeld bewahrheitet. So ist es mit Hilfe der Kollisionsstrategien (z.B. Strategie 4) möglich, die Kraft, welche bei einem Aufprall übertragen wird, zu verringern. Dabei ist jedoch zu beachten, dass je härter das Hindernis und schneller der Manipulator während der Kollision sind, desto schneller wird die Kraft des Aufpralls übertragen. Diese Übertragung geschieht ab einer gewissen Geschwindigkeit so schnell, dass es nach der Kollisionserkennung nicht möglich ist, mit Hilfe der Motoren eine Gegenbewegung zu erzeugen.

Zwar sind den Reaktionsstrategien für solch schnelle Kollisionen klare Grenzen gesetzt, aber dennoch ist es möglich, die Kraft, welche durch die Motor-/Getriebe-Trägheit verzögert in die Kollision wirkt, aufzuheben. Bei Kollisionen, bei denen es möglich ist, die Kollisionskraft zu verringern, bieten die Reaktionsstrategien einen gewissen Schutz. Denn durch die Verringerung der Kollisionskraft werden auch die Momente, welche dadurch in den Gelenken wirken, verringert. Hieraus resultiert, dass die Motoren und Getriebe geschont werden.

Auch welche Gefahr im Allgemeinen von einer Kollision mit dem BioRob ausgeht wurde betrachtet. Überraschenderweise zeigte sich, dass auch bei schnellen Endeffektorgeschwindigkeiten nur sehr geringe Kräfte beim Aufprall auf das Stirnbein bzw. auf den Oberkiefer entstehen. Diese genügen bei weitem nicht, um Frakturen hervorzurufen. Die Gefahr darf jedoch zu diesem Zeitpunkt noch nicht unterschätzt werden, denn weder Kollisionen in eingeklemmter Lage noch mit scharfen Werkzeugen wurden betrachtet und evaluiert. Durch die geschickte Lage der Motoren und das damit verbundene geringe Trägheitsmoment am Endeffektor können jedoch schwere Verletzungen durch stumpfe Einschläge ausgeschlossen werden.

5. Zusammenfassung und Ausblick

Diese Ausarbeitung gibt einen Einblick darüber, welche Sicherheitsprobleme auftreten können, sobald Mensch und Maschine miteinander arbeiten. Ein bestimmtes Niveau an Sicherheit muss gewährleistet sein; denn wer möchte schon gerne eine Arbeitsstelle besuchen, bei der er sich nicht sicher sein kann, ohne Verletzungen diese wieder zu verlassen? Es wurde kurz diskutiert, welche Möglichkeiten der Sicherheitsgewährleistung es gibt. Dabei spielte zumeist eine Vielzahl von Sensoren eine Rolle, die wiederum Geld und Platz am Manipulator erforderten. Aber auch deren Zuverlässigkeit kann an bestimmten Arbeitsumgebungen nicht gesichert werden. Aus diesem Grund bedienten wir uns einer Idee, die fast ohne zusätzliche Sensoren auskommt. Das vorgestellte Verfahren benötigt lediglich einen Positionssensor, um die Gelenkpositionen zu bestimmen. Dann ist es möglich, eine zuverlässige Kollisionserkennung aufzubauen. Über ein Diagnosesignal, welches nur das Positionssignal benötigt, ist es möglich zu ermitteln, ob das angelegte und tatsächliche Motormoment übereinstimmten. Dieses Diagnosesignal wird als Residuum bezeichnet und ist als Vektor dargestellt. Jedes Element des Vektors entspricht dem Residuum eines Gelenks. Das Signal wurde so erstellt das jede Komponente des Vektors die im Gelenk wirkenden Momente aus einer Kollision darstellen.

Damit das gewünschte Verfahren am BioRob eingesetzt werden konnte, musste zunächst das Robotermodell erstellt und schließlich das Residuum passend entwickelt werden. Hierbei musste vor allem die Elastizität der Gelenke am BioRob und die dortige Kraftübertragung betrachtet werden, um das endgültige Robotermodell erstellen zu können. Nachdem auch die Simulationsstrategien vorgestellt wurden, war es an der Simulation mit Matlab/Simulink herauszufinden, welche Möglichkeiten das entwickelte Resiuum für die Kollisionserkennung bot.

Nachdem der Regler zum Abfahren einer Trajektorie erstellt worden war, konnten die ersten Experimente durchgeführt werden. Der Versuchsaufbau wurde dabei sehr einfach gehalten: Der Manipulator kollidierte mit ausgestrecktem Arm mit einem Gegenstand. Durch Änderungen an der Steifheit des Hindernisses sowie der Geschwindigkeiten des Manipulators während der Kollision konnten verschiedene Kollisionszenarien simuliert werden. Nach Betrachtung der Verhalten der einzelnen Strategien konnte festgestellt werden, dass ein reaktives Verhalten nach der Kollisionserkennung am effektivsten ist. Dieses bietet die Möglichkeit, das Residuum zu nutzen, um den Manipulator aus der Kollision automatisch herauszufahren. Somit kann verhindert werden, dass z.B. eine Person vom Manipulator eingeklemmt wird oder die nach der Kollision ansteigende Kraft der weiter drehenden Motoren in die Kollision wirkt.

Des Weiteren konnte gezeigt werden, dass in einem gewissen Rahmen die Kollisionskraft verringert werden kann. Dabei spielt jedoch die Einschlagsgeschwindigkeit aber auch die Steifheit des Hindernisses eine ausschlaggebende Rolle. Gelingt die Verringerung aber, so wird nicht nur die in der Kollision verwickelte Person geschützt, sondern es verringern sich auch die Momente, welche in den Gelenken hervorgerufen werden. Der Roboterarm bzw. die Motoren und Getriebe der Gelenke werden also geschont.

Es folgte abschließend noch die Betrachtung, welche Gefahr im Allgemeinen für den Menschen vom BioRob ausgeht. Beleuchtet wurde dabei ein stumpfer Einschlag des Endeffektors auf die Knochen des menschlichen Schädels. Hier ergaben Experimente mit Endeffektorgeschwindigkeiten von 0.5m/s bis zu 2.5m/s, dass die erzeugte Kollisionskraft bei weitem unter den Werten liegt, die notwendig sind, um eine Fraktur hervorzurufen. Aus dieser Betrachtung ausgelassen sind jedoch die Möglichkeiten, welche Folgen eine Kollision auf den Menschen haben kann, wenn dieser von der Kollision eingeklemmt wird oder ein scharfes Werkzeug am Endeffektor angebracht ist.

Dies führt uns zu den offenen Problemen und Fragestellungen, die in weiteren Arbeiten untersucht werden sollten. Bevor diese eben erwähnten Kollisionen mit z.B. scharfen Gegenständen betrachtet werden können, muss das Modell zunächst auf den Prototypen übertragen werden. Die hier vorgestellten Ergebnisse beziehen sich ausschließlich auf die Simulation mit Matlab/Simulink. Zwar ist das verwendete Robotermodell verifiziert, es bleibt aber offen, wie der Manipulator in Realität reagiert. Auch die Frage, ob das Residuum auf der Roboterplattform implementiert werden kann, ist noch offen. Ist dies gelungen, so können dann die simulierten Kollisionstests durch Messwerte aus der Realität untermauert werden. Zu diesen Messwerten gehören dann vor allem, welche Kollisionskräfte wirklich hervorgerufen werden können und welche wirkliche Gefahr für den Menschen besteht. Aber auch die bekannte Schnelligkeit der Kollisionserkennung kann sich durch den CPU-Takt des Roboters verändern.

Literaturverzeichnis

[1] Anjorin, Anthony: *Aufbau und Validierung eines elastischen Manipulatormodells.* 2007

[2] Craig, John C.: *Introduction to Robotics - Mechanics and Control.* 3. Auflage. Pearson Prentice Hall, 2005. – ISBN 0–201–54361–3. – [Kapitel 6.10, S.185ff] Formulating Manipulator Dynamics in Cartesian Space

[3] De Luca, Alin; Haddadin Sami; Hirzinger G. Allesandro; Albu-Schäffer: Collision Detection and Safe Reaction with the DLR-III Lightweight Manipulator Arm. In: *International Conference on Intelligent Robots and Systems* (2003), S. 634–639

[4] De Luca, Raffaella Allesandro; M. Allesandro; Mattone: Actuator Failure Detection and Isolation Using Generalized Momenta. In: *International Conference on Intelligent Robots and Systems* (2003), S. 634–639

[5] Haddadin, Alin; De Luca Allesandro; Hirzinger G. Sami; Albu-Schäffer: Collision Detection and Reaction: A Contribution to Safe Physical Human-Robot Interaction.

[6] Haddadin, Alin; Hirzinger G. Sami; Albu-Schäffer: Approaching Asimov's 1st Law: the „Impact" of the Robot's Weight Class.

[7] Haddadin, Alin; Hirzinger G. Sami; Albu-Schäffer: Safe Physical Human-Robot Interaction: Measurements, Analysis and New Insights.

[8] Haddadin, Alin; Hirzinger G. Sami; Albu-Schäffer: Safety Evaluation of Physical Human-Robot Interaction via Crash-Testing.

[9] Hauger, Werner; Schnell, Walter; Gross, Dietmar; Schröder, Jörg: *Technische Mechanik 3 - Kinetik.* 8. Auflage. Springer, 2005. – ISBN 3–540–22167–0. – [Kapitel 3] Bewegung eines starren Körpers

[10] Hauger, Werner; Schnell, Walter; Gross, Dietmar; Schröder, Jörg: *Technische Mechanik 1 - Kinetik.* 10. Auflage. Springer, 2008. – ISBN 978–3–540–68394–0. – [Kapitel 8.2, S221f] Der Arbeitssatz

[11] Klug, Sebastian ; Lens, Thomas ; Stryk, Oskar von ; Möhl, Bernhard ; Karguth, Andreas: Biologically Inspired Robot Manipulator for New Applications in Automation Engineering. In: *Proceedings of Robotik 2008.* Munich, Germany : VDI Wissensforum GmbH, June 11-12 2008 (VDI-Berichte 2012)

[12] Lunze, Jan: *Automatisierungstechnik*. Oldenbourg Wissenschaftsverlag, 2003. – ISBN 3–486–27430–9. – [Kapitel 10.2.1, S266f], Fehlererkennung mit einem Zustandsbeobachter - Diagnoseaufgabe

[13] Lunze, Jan: *Regelungstechnik 1: Systemtheoretische Grundlagen, Analyse und Entwurf einschleifiger Regelungen*. 7. Auflage. Springer, 2008. – ISBN 3–540–68907–9. – [Kapitel 5.7.1, S176ff], Proportionalglieder - PT-Glied

[14] Spong, Mark W.; Hutchinson, Seth; Vidyasagar, M.: *Robot Modeling and Control*. Wiley, 2006. – ISBN 3–471–64990–2. – [Kapitel 4.10, S148ff] Static Force/Troque Relationships, [Kapitel 7.3, S255ff] Equations of Motion, [Kapitel 7.5, S267ff] Properties of Robot Dynamic Equations

[15] Wolfram: *Wolfram Mathematica Documentation*. – http://reference.wolfram.com/mathematica/guide/Mathematica.html

A. Matrizen der inversen Dynamik

Mass-Matrix

```
%-- Massen der Glieder:
% m1, m2, m3, m4

%-- Trägheiten der Glieder (Diagonalen der Trägheitsmatritzen):
% I1xx, I1yy,  I1zz
% I2xx, I2yy,  I2zz
% I3xx, I3yy,  I3zz
% I4xx, I4yy,  I4zz

%-- Position des Massenschwerpunkts der Glieder.
%-- Im Glied i bezüglich des Koordinatensystems S_{i} (=> {i}^r_{c_i})
% rc1x, rc1y, rc1z
% rc2x, rc2y, rc2z
% rc3x, rc3y, rc3z
% rc4x, rc4y, rc4z

%-- Länge der Glieder
% l1, l2, l3, l4

%-- Festlegen der Matrixgröße
M = eye(4,4);

%-- erste Spalte der Matrix
M(:,1) = [1/2 * (2 * I1yy + I2xx + I2yy + I3xx + I3yy + I4xx + I4yy...
        + l3^2 * (m3 + m4) + l2^2 * (m2 + m3 + m4) + 2 * m1...
        * (rc1x^2 + rc1z^2) + 2 * l2 * m2 * rc2x + m2...
        * (rc2x^2 + rc2y^2 + 2 * rc2z^2) + 2 * l3 * m3 * rc3x...
        + m3 * (rc3x^2 + rc3y^2 + 2 * rc3z^2) + m4...
        * ((l4 + rc4x)^2 + rc4y^2 + 2 * rc4z^2) + (-I2xx...
        + I2yy + l2^2 * (m2 + m3 + m4) + 2 * l2 * m2 * rc2x...
        + m2 * (rc2x - rc2y) * (rc2x + rc2y)) * cos(2 * q2)...
        + 2 * l2 * (l3 * (m3 + m4) + m3 * rc3x) * cos(q3)...
        + (-I3xx + I3yy) * cos(2 * (q2 + q3)) + l3^2 * m3...
        * cos(2 * (q2 + q3)) + l3^2 * m4 * cos(2 * (q2 + q3))...
        + 2 * l3 * m3 * rc3x * cos(2 * (q2 + q3)) + m3 * rc3x^2...
        * cos(2 * (q2 + q3)) - m3 * rc3y^2 * cos(2 * (q2 + q3))...
        + 2 * l2 * l3 * m3 * cos(2 * q2 + q3) + 2 * l2 * l3 * m4...
        * cos(2 * q2 + q3) + 2 * l2 * m3 * rc3x * cos(2 * q2 + q3)...
        + 2 * l3 * l4 * m4 * cos(q4) + 2 * l3 * m4 * rc4x * cos(q4)...
        + 2 * l2 * l4 * m4 * cos(q3 + q4) + 2 * l2 * m4 * rc4x...
        * cos(q3 + q4) + (-I4xx + I4yy) * cos(2 * (q2 + q3 + q4))...
        + l4^2 * m4 * cos(2 * (q2 + q3 + q4)) + 2 * l4 * m4 * rc4x...
```

```
        * cos(2 * (q2 + q3 + q4)) + m4 * rc4x^2 * cos(2...
        * (q2 + q3 + q4)) - m4 * rc4y^2 * cos(2 * (q2 + q3 + q4))...
        + 2 * l2 * l4 * m4 * cos(2 * q2 + q3 + q4) + 2 * l2 * m4...
        * rc4x * cos(2 * q2 + q3 + q4) + 2 * l3 * l4 * m4 * cos(2...
        * (q2 + q3) + q4) + 2 * l3 * m4 * rc4x * cos(2 * (q2 + q3)...
        + q4) - 2 * l2 * m2 * rc2y * sin(2 * q2) - 2 * m2 * rc2x...
        * rc2y * sin(2 * q2) - 2 * l2 * m3 * rc3y * sin(q3)...
        - 2 * l3 * m3 * rc3y * sin(2 * (q2 + q3)) - 2 * m3 * rc3x...
        * rc3y * sin(2 * (q2 + q3)) - 2 * l2 * m3 * rc3y...
        * sin(2 * q2 + q3) - 2 * l3 * m4 * rc4y * sin(q4)...
        - 2 * l2 * m4 * rc4y * sin(q3 + q4) - 2 * l4 * m4 * rc4y...
        * sin(2 * (q2 + q3 + q4)) - 2 * m4 * rc4x * rc4y * sin(2...
        * (q2 + q3 + q4)) - 2 * l2 * m4 * rc4y * sin(2 * q2...
        + q3 + q4) - 2 * l3 * m4 * rc4y * sin(2 * (q2 + q3) + q4)),...
-m2 * rc2y * rc2z * cos(q2) - m3 * rc3y * rc3z * cos(q2 + q3)...
        - (m2 * (l2 + rc2x) * rc2z + l2 * m3 * rc3z) * sin(q2)...
        - m3 * (l3 + rc3x) * rc3z * sin(q2 + q3) - m4 * rc4z...
        * (rc4y * cos(q2 + q3 + q4) + l2 * sin(q2) + l3 * sin(q2 + q3)...
        + (l4 + rc4x) * sin(q2 + q3 + q4)),...
-m3 * rc3y * rc3z * cos(q2 + q3) - m3 * (l3 + rc3x) * rc3z...
        * sin(q2 + q3) - m4 * rc4z * (rc4y * cos(q2 + q3 + q4) + l3...
        * sin(q2 + q3) + (l4 + rc4x) * sin(q2 + q3 + q4)),...
-m4 * rc4z * (rc4y * cos(q2 + q3 + q4) + (l4 + rc4x)...
        * sin(q2 + q3 + q4))];

%-- zweite Spalte der Matrix
M(:,2) = [-m2 * rc2y * rc2z * cos(q2) - m3 * rc3y * rc3z...
        * cos(q2 + q3)  - (m2 * (l2 + rc2x) * rc2z + l2 * m3 * rc3z)...
        * sin(q2) - m3 * (l3 + rc3x) * rc3z * sin(q2 + q3) - m4...
        * rc4z * (rc4y * cos(q2 + q3 + q4) + l2 * sin(q2) + l3...
        * sin(q2 + q3) + (l4 + rc4x) * sin(q2 + q3 + q4)),...
I2zz + I3zz + I4zz + l3^2 * (m3 + m4) + l2^2 * (m2 + m3 + m4)...
        + 2 * l2 * m2 * rc2x + m2 * (rc2x^2 + rc2y^2) + 2 * l3...
        * m3 * rc3x + m3 * (rc3x^2 + rc3y^2) + m4...
        * ((l4 + rc4x)^2 + rc4y^2) + 2 * (l2 * (l3 * (m3 + m4)...
        + m3 * rc3x) * cos(q3) + l3 * m4 * (l4 + rc4x) * cos(q4)...
        + l2 * l4 * m4 * cos(q3 + q4) + l2 * m4 * rc4x...
        * cos(q3 + q4) - l2 * m3 * rc3y * sin(q3) - l3 * m4...
        * rc4y * sin(q4) - l2 * m4 * rc4y * sin(q3 + q4)),...
I3zz + I4zz + l3^2 * (m3 + m4) + 2 * l3 * m3 * rc3x + m3...
        * (rc3x^2 + rc3y^2) + m4 * ((l4 + rc4x)^2 + rc4y^2) + l2...
        * (l3 * (m3 + m4) + m3 * rc3x) * cos(q3) + 2 * l3 * m4...
        * (l4 + rc4x) * cos(q4) + l2 * l4 * m4 * cos(q3 + q4) + l2...
        * m4 * rc4x * cos(q3 + q4) - l2 * m3 * rc3y * sin(q3) - 2...
        * l3 * m4 * rc4y * sin(q4) - l2 * m4 * rc4y * sin(q3 + q4),...
I4zz + m4 * ((l4 + rc4x)^2 + rc4y^2) + l3 * m4 * (l4 + rc4x)...
        * cos(q4) + l2 * m4 * (l4 + rc4x) * cos(q3 + q4)...
        - m4 * rc4y * (l3 * sin(q4) + l2 * sin(q3 + q4))];

%-- dritte Spalte der Matrix
M(:,3) = [-m3 * rc3y * rc3z * cos(q2 + q3) - m3 * (l3 + rc3x) * rc3z...
        * sin(q2 + q3) - m4 * rc4z * (rc4y * cos(q2 + q3 + q4) + l3...
        * sin(q2 + q3) + (l4 + rc4x) * sin(q2 + q3 + q4)),...
I3zz + I4zz + l3^2 * (m3 + m4) + 2 * l3 * m3 * rc3x + m3...
        * (rc3x^2 + rc3y^2) + m4 * ((l4 + rc4x)^2 + rc4y^2) + l2...
```

```
        * (l3 * (m3 + m4) + m3 * rc3x) * cos(q3) + 2 * l3 * m4...
        * (l4 + rc4x) * cos(q4) + l2 * l4 * m4 * cos(q3 + q4) + l2...
        * m4 * rc4x * cos(q3 + q4) - l2 * m3 * rc3y * sin(q3) - 2...
        * l3 * m4 * rc4y * sin(q4) - l2 * m4 * rc4y * sin(q3 + q4),...
I3zz + I4zz + l3^2 * (m3 + m4) + 2 * l3 * m3 * rc3x + m3...
        * (rc3x^2 + rc3y^2) + m4 * ((l4 + rc4x)^2 + rc4y^2)...
        + 2 * l3 * m4 * (l4 + rc4x) * cos(q4) - 2 * l3...
        * m4 * rc4y * sin(q4),...
I4zz + m4 * ((l4 + rc4x)^2 + rc4y^2) + l3 * m4 * (l4 + rc4x)...
        * cos(q4) - l3 * m4 * rc4y * sin(q4)];

%-- vierte Spalte der Matrix
M(:,4) = [-m4 * rc4z * (rc4y * cos(q2 + q3 + q4) + (l4 + rc4x)...
        * sin(q2 + q3 + q4)),...
I4zz + m4 * ((l4 + rc4x)^2 + rc4y^2) + l3 * m4 * (l4 + rc4x)...
        * cos(q4) + l2 * m4 * (l4 + rc4x) * cos(q3 + q4)...
        - m4 * rc4y * (l3 * sin(q4) + l2 * sin(q3 + q4)),...
I4zz + m4 * ((l4 + rc4x)^2 + rc4y^2) + l3 * m4 * (l4 + rc4x)...
        * cos(q4) - l3 * m4 * rc4y * sin(q4),...
I4zz + m4 * ((l4 + rc4x)^2 + rc4y^2)];
```

Coriolis-Matrix

```
%-- Massen der Glieder:
% m1, m2, m3, m4

%-- Trägheiten der Glieder (Diagonalen der Trägheitsmatritzen):
% I1xx, I1yy,  I1zz
% I2xx, I2yy,  I2zz
% I3xx, I3yy,  I3zz
% I4xx, I4yy,  I4zz

%-- Position des Massenschwerpunkts der Glieder.
%-- Im Glied i bezüglich des Koordinatensystems S_{i} (=> {i}^r_{c_i})
% rc1x, rc1y, rc1z
% rc2x, rc2y, rc2z
% rc3x, rc3y, rc3z
% rc4x, rc4y, rc4z

%-- Länge der Glieder
% l1, l2, l3, l4

%-- Festlegen der Matrixgröße
C = eye(4,4);

%-- erste Spalte der Matrix
C(1,:) = [1/2 * (-(2 * m2 * (l2 + rc2x) * rc2y * cos(2 * q2) + 2...
        * m3 * (l3 + rc3x) * rc3y * cos(2 * (q2 + q3)) + 2 * l2...
        * m3 * rc3y * cos(2 * q2 + q3) + 2 * m4 * (l4 + rc4x)...
        * rc4y * cos(2 * (q2 + q3 + q4)) + 2 * l2 * m4 * rc4y...
        * cos(2 * q2 + q3 + q4) + 2 * l3 * m4 * rc4y...
```

```
         * cos(2 * (q2 + q3) + q4) + (-I2xx + I2yy) * sin(2 * q2)...
         + l2^2 * m2 * sin(2 * q2) + l2^2 * m3 * sin(2 * q2)...
         + l2^2 * m4 * sin(2 * q2) + 2 * l2 * m2 * rc2x...
         * sin(2 * q2) + m2 * rc2x^2 * sin(2 * q2) - m2 * rc2y^2...
         * sin(2 * q2) + (-I3xx + I3yy) * sin(2 * (q2 + q3))...
         + l3^2 * m3 * sin(2 * (q2 + q3)) + l3^2 * m4...
         * sin(2 * (q2 + q3)) + 2 * l3 * m3 * rc3x...
         * sin(2 * (q2 + q3)) + m3 * rc3x^2 * sin(2 * (q2 + q3))...
         - m3 * rc3y^2 * sin(2 * (q2 + q3)) + 2 * l2 * l3 * m3...
         * sin(2 * q2 + q3) + 2 * l2 * l3 * m4 * sin(2 * q2 + q3)...
         + 2 * l2 * m3 * rc3x * sin(2 * q2 + q3) + (-I4xx + I4yy)...
         * sin(2 * (q2 + q3 + q4)) + l4^2 * m4...
         * sin(2 * (q2 + q3 + q4)) + 2 * l4 * m4 * rc4x..
         * sin(2 * (q2 + q3 + q4)) + m4 * rc4x^2...
         * sin(2 * (q2 + q3 + q4)) - m4 * rc4y^2...
         * sin(2 * (q2 + q3 + q4)) + 2 * l2 * l4 * m4...
         * sin(2 * q2 + q3 + q4) + 2 * l2 * m4 * rc4x...
         * sin(2 * q2 + q3 + q4) + 2 * l3 * m4 * (l4 + rc4x)...
         * sin(2 * (q2 + q3) + q4)) * q2_d - (l2 * m3 * rc3y...
         * cos(q3) + 2 * m3 * (l3 + rc3x) * rc3y...
         * cos(2 * (q2 + q3)) + l2 * m3 * rc3y * cos(2 * q2 + q3)...
         + l2 * m4 * rc4y * cos(q3 + q4) + 2 * l4 * m4 * rc4y...
         * cos(2 * (q2 + q3 + q4)) + 2 * m4 * rc4x * rc4y...
         * cos(2 * (q2 + q3 + q4)) + l2 * m4 * rc4y * cos(2 * q2...
         + q3 + q4) + 2 * l3 * m4 * rc4y * cos(2 * (q2 + q3) + q4)...
         + l2 * l3 * m3 * sin(q3) + l2 * l3 * m4 * sin(q3) + l2...
         * m3 * rc3x * sin(q3) + (-I3xx + I3yy) * sin(2 * (q2...
         + q3)) + l3^2 * m3 * sin(2 * (q2 + q3)) + l3^2 * m4...
         * sin(2 * (q2 + q3)) + 2 * l3 * m3 * rc3x * sin(2...
         * (q2 + q3)) + m3 * rc3x^2 * sin(2 * (q2 + q3)) - m3...
         * rc3y^2 * sin(2 * (q2 + q3)) + l2 * l3 * m3 * sin(2...
         * q2 + q3) + l2 * l3 * m4 * sin(2 * q2 + q3) + l2 * m3...
         * rc3x * sin(2 * q2 + q3) + l2 * l4 * m4 * sin(q3 + q4)...
         + l2 * m4 * rc4x * sin(q3 + q4) + (-I4xx + I4yy)...
         * sin(2 * (q2 + q3 + q4)) + l4^2 * m4 * sin(2 * (q2 + q3...
         + q4)) + 2 * l4 * m4 * rc4x * sin(2 * (q2 + q3 + q4))...
         + m4 * rc4x^2 * sin(2 * (q2 + q3 + q4)) - m4 * rc4y^2...
         * sin(2 * (q2 + q3 + q4)) + l2 * l4 * m4 * sin(2 * q2...
         + q3 + q4) + l2 * m4 * rc4x * sin(2 * q2 + q3 + q4)...
         + 2 * l3 * m4 * (l4 + rc4x) * sin(2 * (q2 + q3) + q4))...
         * q3_d - (2 * m4 * (rc4y * cos(q2 + q3 + q4) + (l4 + rc4x)...
         * sin(q2 + q3 + q4)) * (l2 * cos(q2) + l3 * cos(q2 + q3)...
         + (l4 + rc4x) * cos(q2 + q3 + q4) - rc4y * sin(q2...
         + q3 + q4)) - (I4xx - I4yy) * sin(2 * (q2 + q3...
         + q4))) * q4_d) ,...
1/2 * (-(2 * m2 * (l2 + rc2x) * rc2y * cos(2 * q2) + 2 * m3...
         * (l3 + rc3x) * rc3y * cos(2 * (q2 + q3)) + 2 * l2 * m3...
         * rc3y * cos(2 * q2 + q3) + 2 * m4 * (l4 + rc4x) * rc4y...
         * cos(2 * (q2 + q3 + q4)) + 2 * l2 * m4 * rc4y * cos(2 * q2...
         + q3 + q4) + 2 * l3 * m4 * rc4y * cos(2 * (q2 + q3) + q4)...
         + (-I2xx + I2yy) * sin(2 * q2) + l2^2 * m2 * sin(2 * q2)...
         + l2^2 * m3 * sin(2 * q2) + l2^2 * m4 * sin(2 * q2)...
         + 2 * l2 * m2 * rc2x * sin(2 * q2) + m2 * rc2x^2...
         * sin(2 * q2) - m2 * rc2y^2 * sin(2 * q2) + (-I3xx + I3yy)...
         * sin(2 * (q2 + q3)) + l3^2 * m3 * sin(2 * (q2 + q3))...
```

```
        + l3^2 * m4 * sin(2 * (q2 + q3)) + 2 * l3 * m3 * rc3x...
        * sin(2 * (q2 + q3)) + m3 * rc3x^2 * sin(2 * (q2 + q3))...
        - m3 * rc3y^2 * sin(2 * (q2 + q3)) + 2 * l2 * l3 * m3...
        * sin(2 * q2 + q3) + 2 * l2 * l3 * m4 * sin(2 * q2 + q3)...
        + 2 * l2 * m3 * rc3x * sin(2 * q2 + q3) + (-I4xx + I4yy)...
        * sin(2 * (q2 + q3 + q4)) + l4^2 * m4 * sin(2 * (q2 + q3...
        + q4)) + 2 * l4 * m4 * rc4x * sin(2 * (q2 + q3 + q4))...
        + m4 * rc4x^2 * sin(2 * (q2 + q3 + q4)) - m4 * rc4y^2...
        * sin(2 * (q2 + q3 + q4)) + 2 * l2 * l4 * m4 * sin(2 * q2...
        + q3 + q4) + 2 * l2 * m4 * rc4x * sin(2 * q2 + q3 + q4)...
        + 2 * l3 * m4 * (l4 + rc4x) * sin(2 * (q2 + q3) + q4))...
        * q1_d - 2 * (((m2 * rc2x * rc2z + l2 * (m2 * rc2z + m3...
        * rc3z + m4 * rc4z)) * cos(q2) + (m3 * (l3 + rc3x)...
        * rc3z + l3 * m4 * rc4z) * cos(q2 + q3) - m2 * rc2y...
        * rc2z * sin(q2) - m3 * rc3y * rc3z * sin(q2 + q3)...
        + m4 * rc4z * ((l4 + rc4x) * cos(q2 + q3 + q4) - rc4y...
        * sin(q2 + q3 + q4))) * q2_d + ((m3 * (l3 + rc3x)...
        * rc3z + l3 * m4 * rc4z) * cos(q2 + q3) + m4 * (l4...
        + rc4x) * rc4z * cos(q2 + q3 + q4) - m3 * rc3y * rc3z...
        * sin(q2 + q3) - m4 * rc4y * rc4z * sin(q2 + q3 + q4))...
        * q3_d + m4 * rc4z * ((l4 + rc4x) * cos(q2 + q3 + q4)...
        - rc4y * sin(q2 + q3 + q4)) * q4_d)) ,...
1/2 * (-(l2 * m3 * rc3y * cos(q3) + 2 * m3 * (l3 + rc3x) * rc3y...
        * cos(2 * (q2 + q3)) + l2 * m3 * rc3y * cos(2 * q2 + q3)...
        + l2 * m4 * rc4y * cos(q3 + q4) + 2 * l4 * m4 * rc4y...
        * cos(2 * (q2 + q3 + q4)) + 2 * m4 * rc4x * rc4y...
        * cos(2 * (q2 + q3 + q4)) + l2 * m4 * rc4y * cos(2 * q2...
        + q3 + q4) + 2 * l3 * m4 * rc4y * cos(2 * (q2 + q3) + q4)...
        + l2 * l3 * m3 * sin(q3) + l2 * l3 * m4 * sin(q3)...
        + l2 * m3 * rc3x * sin(q3) + (-I3xx + I3yy) * sin(2...
        * (q2 + q3)) + l3^2 * m3 * sin(2 * (q2 + q3)) + l3^2...
        * m4 * sin(2 * (q2 + q3)) + 2 * l3 * m3 * rc3x * sin(2...
        * (q2 + q3)) + m3 * rc3x^2 * sin(2 * (q2 + q3)) - m3...
        * rc3y^2 * sin(2 * (q2 + q3)) + l2 * l3 * m3 * sin(2...
        * q2 + q3) + l2 * l3 * m4 * sin(2 * q2 + q3) + l2 * m3...
        * rc3x * sin(2 * q2 + q3) + l2 * l4 * m4 * sin(q3 + q4)...
        + l2 * m4 * rc4x * sin(q3 + q4) + (-I4xx + I4yy) * sin(2...
        * (q2 + q3 + q4)) + l4^2 * m4 * sin(2 * (q2 + q3 + q4))...
        + 2 * l4 * m4 * rc4x * sin(2 * (q2 + q3 + q4)) + m4...
        * rc4x^2 * sin(2 * (q2 + q3 + q4)) - m4 * rc4y^2...
        * sin(2 * (q2 + q3 + q4)) + l2 * l4 * m4 * sin(2 * q2...
        + q3 + q4) + l2 * m4 * rc4x * sin(2 * q2 + q3 + q4)...
        + 2 * l3 * m4 * (l4 + rc4x) * sin(2 * (q2 + q3) + q4))...
        * q1_d - 2 * ((m3 * (l3 + rc3x) * rc3z + l3 * m4 * rc4z)...
        * cos(q2 + q3) + m4 * (l4 + rc4x) * rc4z * cos(q2...
        + q3 + q4) - m3 * rc3y * rc3z * sin(q2 + q3) - m4...
        * rc4y * rc4z * sin(q2 + q3 + q4)) * (q2_d + q3_d)...
        - 2 * m4 * rc4z * ((l4 + rc4x) * cos(q2 + q3 + q4)...
        - rc4y * sin(q2 + q3 + q4)) * q4_d) ,...
1/2 * (-(2 * m4 * (rc4y * cos(q2 + q3 + q4) + (l4 + rc4x)...
        * sin(q2 + q3 + q4)) * (l2 * cos(q2) + l3 * cos(q2 + q3)...
        + (l4 + rc4x) * cos(q2 + q3 + q4) - rc4y...
        * sin(q2 + q3 + q4)) - (I4xx - I4yy) * sin(2...
        * (q2 + q3 + q4))) * q1_d - 2 * m4 * rc4z * ((l4 + rc4x)...
        * cos(q2 + q3 + q4) - rc4y * sin(q2 + q3 + q4))...
```

```
        * (q2_d + q3_d + q4_d))];

%-- zweite Spalte der Matrix
C(2,:) = [1/2 * (2 * m2 * (l2 + rc2x) * rc2y * cos(2 * q2)...
        + 2 * m3 * (l3 + rc3x) * rc3y * cos(2 * (q2 + q3))...
        + 2 * l2 * m3 * rc3y * cos(2 * q2 + q3) + 2 * m4...
        * (l4 + rc4x) * rc4y * cos(2 * (q2 + q3 + q4))...
        + 2 * l2 * m4 * rc4y * cos(2 * q2 + q3 + q4)...
        + 2 * l3 * m4 * rc4y * cos(2 * (q2 + q3) + q4)...
        + (-I2xx + I2yy) * sin(2 * q2) + l2^2 * m2...
        * sin(2 * q2) + l2^2 * m3 * sin(2 * q2) + l2^2 * m4...
        * sin(2 * q2) + 2 * l2 * m2 * rc2x * sin(2 * q2)...
        + m2 * rc2x^2 * sin(2 * q2) - m2 * rc2y^2 * sin(2 * q2)...
        + (-I3xx + I3yy) * sin(2 * (q2 + q3)) + l3^2 * m3...
        * sin(2 * (q2 + q3)) + l3^2 * m4 * sin(2 * (q2 + q3))...
        + 2 * l3 * m3 * rc3x * sin(2 * (q2 + q3)) + m3 * rc3x^2...
        * sin(2 * (q2 + q3)) - m3 * rc3y^2 * sin(2 * (q2 + q3))...
        + 2 * l2 * l3 * m3 * sin(2 * q2 + q3) + 2 * l2 * l3...
        * m4 * sin(2 * q2 + q3) + 2 * l2 * m3 * rc3x...
        * sin(2 * q2 + q3) + (-I4xx + I4yy) * sin(2 * (q2...
        + q3 + q4)) + l4^2 * m4 * sin(2 * (q2 + q3 + q4))...
        + 2 * l4 * m4 * rc4x * sin(2 * (q2 + q3 + q4)) + m4...
        * rc4x^2 * sin(2 * (q2 + q3 + q4)) - m4 * rc4y^2...
        * sin(2 * (q2 + q3 + q4)) + 2 * l2 * l4 * m4 * sin(2...
        * q2 + q3 + q4) + 2 * l2 * m4 * rc4x * sin(2 * q2...
        + q3 + q4) + 2 * l3 * m4 * (l4 + rc4x) * sin(2...
        * (q2 + q3) + q4)) * q1_d ,...
-l2 * (m3 * rc3y * cos(q3) + m4 * rc4y * cos(q3 + q4) + (l3...
        * (m3 + m4) + m3 * rc3x) * sin(q3) + m4 * (l4 + rc4x)...
        * sin(q3 + q4)) * q3_d - m4 * (l3 * rc4y * cos(q4)...
        + l2 * rc4y * cos(q3 + q4) + (l4 + rc4x) * (l3...
        * sin(q4) + l2 * sin(q3 + q4))) * q4_d ,...
-l2 * (m3 * rc3y * cos(q3) + m4 * rc4y * cos(q3 + q4) + (l3...
        * (m3 + m4) + m3 * rc3x) * sin(q3) + m4 * (l4 + rc4x)...
        * sin(q3 + q4)) * (q2_d + q3_d) - m4 * (l3 * rc4y...
        * cos(q4) + l2 * rc4y * cos(q3 + q4) + (l4 + rc4x)...
        * (l3 * sin(q4) + l2 * sin(q3 + q4))) * q4_d ,...
-m4 * (l3 * rc4y * cos(q4) + l2 * rc4y * cos(q3 + q4)...
        + (l4 + rc4x) * (l3 * sin(q4) + l2 * sin(q3 + q4)))...
        * (q2_d + q3_d + q4_d)];

%-- dritte Spalte der Matrix
C(3,:) = [1/2 * (l2 * m3 * rc3y * cos(q3) + 2 * m3...
        * (l3 + rc3x) * rc3y * cos(2 * (q2 + q3)) + l2 * m3...
        * rc3y * cos(2 * q2 + q3) + l2 * m4 * rc4y...
        * cos(q3 + q4) + 2 * l4 * m4 * rc4y * cos(2 * (q2...
        + q3 + q4)) + 2 * m4 * rc4x * rc4y * cos(2 * (q2...
        + q3 + q4)) + l2 * m4 * rc4y * cos(2 * q2 + q3 + q4)...
        + 2 * l3 * m4 * rc4y * cos(2 * (q2 + q3) + q4)...
        + l2 * l3 * m3 * sin(q3) + l2 * l3 * m4 * sin(q3)...
        + l2 * m3 * rc3x * sin(q3) + (-I3xx + I3yy)...
        * sin(2 * (q2 + q3)) + l3^2 * m3 * sin(2...
        * (q2 + q3)) + l3^2 * m4 * sin(2 * (q2 + q3))...
        + 2 * l3 * m3 * rc3x * sin(2 * (q2 + q3)) + m3...
        * rc3x^2 * sin(2 * (q2 + q3)) - m3 * rc3y^2...
```

```
        * sin(2 * (q2 + q3)) + l2 * l3 * m3 * sin(2...
        * q2 + q3) + l2 * l3 * m4 * sin(2 * q2 + q3)...
        + l2 * m3 * rc3x * sin(2 * q2 + q3) + l2 * l4 * m4...
        * sin(q3 + q4) + l2 * m4 * rc4x * sin(q3 + q4)...
        + (-I4xx + I4yy) * sin(2 * (q2 + q3 + q4)) + l4^2...
        * m4 * sin(2 * (q2 + q3 + q4)) + 2 * l4 * m4 * rc4x...
        * sin(2 * (q2 + q3 + q4)) + m4 * rc4x^2 * sin(2...
        * (q2 + q3 + q4)) - m4 * rc4y^2 * sin(2 * (q2...
        + q3 + q4)) + l2 * l4 * m4 * sin(2 * q2 + q3 + q4)...
        + l2 * m4 * rc4x * sin(2 * q2 + q3 + q4) + 2 * l3...
        * m4 * (l4 + rc4x) * sin(2 * (q2 + q3) + q4)) * q1_d ,...
l2 * (m3 * rc3y * cos(q3) + m4 * rc4y * cos(q3 + q4)...
        + (l3 * (m3 + m4) + m3 * rc3x) * sin(q3) + m4...
        * (l4 + rc4x) * sin(q3 + q4)) * q2_d - l3 * m4...
        * (rc4y * cos(q4) + (l4 + rc4x) * sin(q4)) * q4_d ,...
-l3 * m4 * (rc4y * cos(q4) + (l4 + rc4x) * sin(q4)) * q4_d ,...
-l3 * m4 * (rc4y * cos(q4) + (l4 + rc4x) * sin(q4))...
        * (q2_d + q3_d + q4_d)];

%-- vierte Spalte der Matrix
C(4,:) = [1/2 * (-2 * m4 * (rc4y * cos(q2 + q3 + q4)...
        + (l4 + rc4x) * sin(q2 + q3 + q4)) * (-l2 * cos(q2)...
        - l3 * cos(q2 + q3) - (l4 + rc4x) * cos(q2 + q3 + q4)...
        + rc4y * sin(q2 + q3 + q4)) + (-I4xx + I4yy)...
        * sin(2 * (q2 + q3 + q4))) * q1_d ,...
m4 * ((l3 * rc4y * cos(q4) + l2 * rc4y * cos(q3 + q4)...
        + (l4 + rc4x) * (l3 * sin(q4) + l2 * sin(q3 + q4)))...
        * q2_d + l3 * (rc4y * cos(q4) + (l4 + rc4x)...
        * sin(q4)) * q3_d) ,...
l3 * m4 * (rc4y * cos(q4) + (l4 + rc4x)...
        * sin(q4)) * (q2_d + q3_d) ,...
0];
```

Gravitations-Vektor

```
%-- Massen der Glieder:
% m1, m2, m3, m4

%-- Trägheiten der Glieder (Diagonalen der Trägheitsmatritzen):
% I1xx, I1yy,  I1zz
% I2xx, I2yy,  I2zz
% I3xx, I3yy,  I3zz
% I4xx, I4yy,  I4zz

%-- Position des Massenschwerpunkts der Glieder.
%-- Im Glied i bezüglich des Koordinatensystems S_{i} (=> {i}^r_{c_i})
% rc1x, rc1y, rc1z
% rc2x, rc2y, rc2z
% rc3x, rc3y, rc3z
% rc4x, rc4y, rc4z

```

```
%-- Länge der Glieder
% l1, l2, l3, l4


%-- erste Zeile des Vektors
g = [0;...

%-- zweite Zeile des Vektors
 -g1 * ((l2 * (m2 + m3 + m4) + m2 * rc2x) * cos(q2) + (l3 * (m3...
        + m4) + m3 * rc3x) * cos(q2 + q3) + m4 * (l4 + rc4x)...
        * cos(q2 + q3 + q4) - m2 * rc2y * sin(q2) - m3 * rc3y...
        * sin(q2 + q3) - m4 * rc4y * sin(q2 + q3 + q4));...
%-- dritte Zeile des Vektors
 -g1 * ((l3 * (m3 + m4) + m3 * rc3x) * cos(q2 + q3) + m4...
        * (l4 + rc4x) * cos(q2 + q3 + q4) - m3 * rc3y...
        * sin(q2 + q3) - m4 * rc4y * sin(q2 + q3 + q4));...
%-- vierte Zeile des Vektors
 g1 * m4 * (-(l4 + rc4x) * cos(q2 + q3 + q4)...
        + rc4y * sin(q2 + q3 + q4))];
```

Reibungs-Vektor

```
F = [d1*q1_d; d2*q2_d; d3*q3_d; d4*q4_d];
```

B. Aufbau der Matlabsimulation

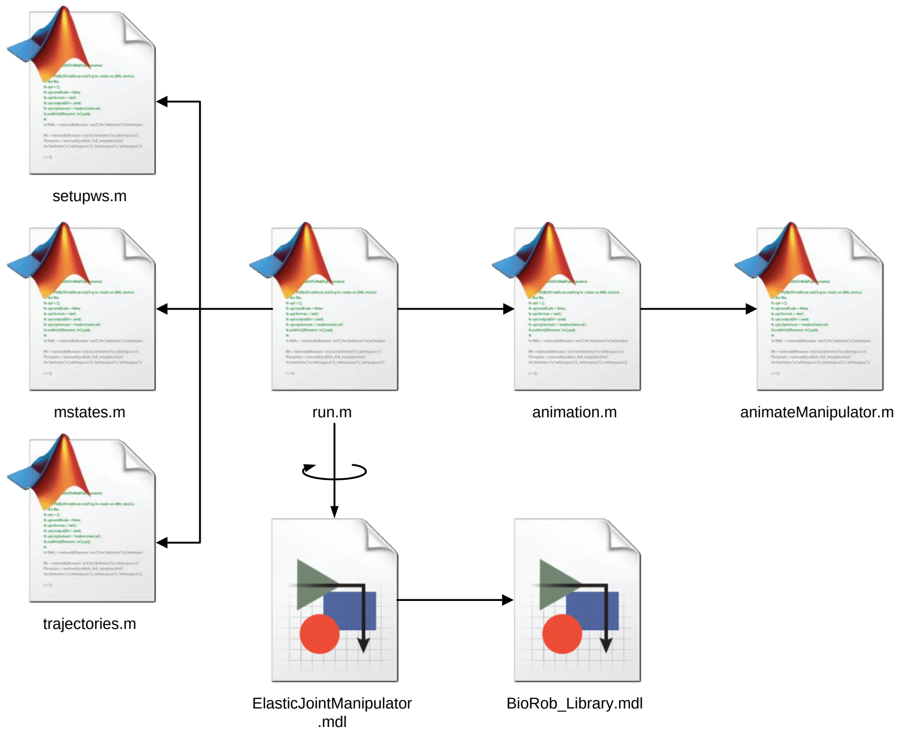


Bild B.1.: Übersicht über die Dateien, welche an der Simulation beteiligt sind

Wie in Bild B zu sehen, ist das Matlabprojekt in mehrere Dateien aufgeteilt. Die Simulation wird von der Hauptdatei, der „run.m„ gesteuert. Diese ruft als vorbereitende Maßnahme für die Simulation die Dateien „setupws.m“, „mstates.m“ sowie „trajectory.m“ auf. In der Datei „setupws.m“ werden alle den Manipulator beschreibenden Parameter geladen (z.B. Länge der Glieder, Trägheitsmatrizen, usw.). Der initiale Modellzustand (z.B. Motor- und Gelenkpositionen zum Zeitpunkt $t = 0$) werden in der Datei „mstates.m“ definiert. Damit eine Trajektorie abgefahren werden kann, muss diese zunächst einmal erstellt werden. Dies geschieht in der Datei „trajectory.m“. Sind diese vorbereitenden Maßnahmen abgeschlossen, kann die Simulation durchgeführt werden. Hierbei simuliert Matlab den in der Modelldatei „elasticJointManipulator.mdl“ erstellten Manipulator. Dieser greift auch auf die z.B. in der Datei „setupws.m“ definierten Werte zurück. Das Modell selbst ist unter anderem aus Blöcken zusammengestellt, welche in einer eigenen Bibliothek zusammengefasst wurden. Daher ist es für die Simulation zwingend notwendig, dass diese Bibliothek dem Modell zur Verfügung steht. Ist die Simulation durchgelaufen, wird eine Animation, gesteuert von der Datei „animation.m“, angezeigt. Auch Plots aus den Daten der Simulation können hier visualisiert werden.

Um konkret die Simulation starten zu können, muss die Datei „run.m“ in Matlab geöffnet werden. Nun muss die erste Zelle ausgeführt werden (Klick in die erste Zeile des Datei, Strg + Return betätigen), damit die für die Simulation notwendigen Parameter in den Arbeitsraum geladen werden. Im Anschluss mit F5 die Datei komplett ausführen. Nachdem die Simulation erfolgreich durchgelaufen ist, erscheint automatisch die Visualisierung der Simulationsergebnisse.