Entwicklung eines Systems zur echtzeitfähigen Abbildung der Bewegung eines menschlichen Arms auf einen Roboterarm

Gökçe Aydos

20. März 2011

Erklärung

Ich versichere, die vorliegende Arbeit selbständig und nur unter Benutzung der angegebenen Hilfsmittel angefertigt zu haben.

Aachen, den 20. März 2011

Inhaltsverzeichnis

1 Einleitung und Aufgabenstellung **3**

1.1 Einleitung 3

1.2 Aufgabenstellung 5

2 Benutzte Hardware und Software **7**

2.1 Kuka LBR IV 7

2.2 Vicon 3D-Tracking-System 9

3 Systemaufbau **13**

3.1 Allgemeiner Systemaufbau 13

3.2 Realisierung der Module 14

4 Validierung **21**

5 Zusammenfassung und Ausblick **23**

Kapitel 1

Einleitung und Aufgabenstellung

1.1 Einleitung

Viele Menschen leiden unter Bewegungsstörungen, die durch physiotherapeutische Behandlung rehabilitiert werden. Viele dieser Patienten sind die Schlaganfall-Erkrankte.

Bei Schlaganfall handelt es sich um eine plötzlich auftretende Erkrankung des Gehirns, die oft zu einem anhaltenden Ausfall der Funktionen des Zentralnervensystems führt. Die Gründe dafür sind vielfältig, verursacht werden kann ein Schlaganfall z.B. durch Arterienverkalkung, Gefäßmißbildungen oder Herzfehler. Diese führen dann zu einer kritischen Blutversorgung des Gehirns, was wiederum einen Sauerstoffmangel bei den betroffenen Nervenzellen einleitet und zu deren Tod führen kann. Abhängig vom betroffenen Gehirnareal können sämtliche Teile des Körpers nicht richtig oder gar nicht gesteuert werden, was zu deren Ausfall und auch zum Tod führt. Dabei wird die Anzahl von Schlaganfall-Patienten in Deutschland jährlich auf 250 000[1] geschätzt, wobei die jährlichen Behandlungs- und Pflegekosten aller Schlaganfälle etwa zwei Prozent der Ausgaben der gesetzlichen Krankenversicherung ausmachen.

Die überlebenden Patienten haben als Folge des Schlaganfalls Sprach-, Schluck-, Bewusstseins-, Seh-, Wahrnehmungsstörungen sowie Lähmungen und Gefühlsstörungen der Arme und Beine. Wie oben beschrieben entstehen diese durch den Ausfall von entsprechenden Gehirnarealen, so dass der Patient dadurch die Bewegung nicht ausführen kann, für die dieses Gehirnareal verantwortlich ist.

[1] Quelle: `http://www.schlaganfall-hilfe.de/images/stories/sdsh/presse/faq/2010_okt_zahlen_daten_fakten.pdf`

Nichtsdestotrotz hat das Gehirn die Eigenschaft, Areale je nach Nutzung in Größe und Antworteigenschaft zu variieren. So ist z.B. ein Teil des Gehirns, der normalerweise für die Bewegung der Hand verantwortlich ist, in der Lage, auch benachbarte Orte wie den Ellenbogen zu steuern. Das ermöglicht die Kompensation der durch einen Schlaganfall ausgefallenen Areale des Gehirns durch andere gesunde Areale. Diese Eigenschaft kann man inbesonders bei der Behandlung von Lähmungen benutzen. Dabei werden alltägliche Bewegungsabläufe, die durch den Schlaganfall verloren gegangen sind, so lange wiederholt, bis das Gehirn die verlorenen Bewegungsabläufe wieder beherrscht.

Diese Übungen werden meistens durch physiotherapeutisches Fachpersonal vorgeführt, betreut und überwacht. Leider nimmt diese Art der Behandlung viel Zeit in Anspruch und ist kostenintensiv, wodurch die erforderlichen Übungseinheiten nur unzureichend ausgeführt werden können.

Um diesem Problem entgegenzuwirken, können Armroboter eingesetzt werden, die den repetitiven und zeitintensiven Teil dieser Übungen übernehmen und so den Physiotherapeuten entlasten. Dabei gibt der Roboter mit seinem TCP[2] dem Patienten die Bewegungsbahn so vor, wie der Therapeut den Patienten führen würde.

Dabei spielt die Programmierung der Robotertrajektorie eine wichtige Rolle, weil der genaue Bewegungsablauf kritisch für den Erfolg der Therapie ist. Aus diesem Grund erfolgt die Programmierung des Roboters meistens simultan, d.h. der Roboter wird während der Führung des Patienten auch mitgeführt. Das wiederum erfordert ungewohnte Fähigkeiten vom Therapeuten, da er auch auf die richtige Stellung der Roboterachsen achten muss.

Ein ähnliches Problem taucht auf, wenn man einen anderen Roboter mit verschiedenartigen Gelenken verwendet, wie z.B. ein Roboter, der manche Zielpositionen nur in bestimmten Gelenkpositionen anfahren kann. In diesem Fall muss der Therapeut wissen, in welchen Gelenkstellungen der Roboter welche Positionen erreichen kann, damit die gewünschte Trajektorie durch den Roboter auch realisierbar ist. In diesem Fall wäre es wünschenswert, wenn die Programmierung der Trajektorie unabhängig vom Roboter erfolgen würde. D.h. die gewünschte Trajektorie wird einmal aufgenommen, in einem Rechner werden die Gelenkpositionen für den benutzten Roboter berechnet und schließlich zum Roboter übertragen.

[2]Tool Center Point, Werkzeugposition eines Roboters

1.2 Aufgabenstellung

Ein möglicher Lösungsansatz für die oben genannten Probleme ist der Einsatz eines Systems, das die Bewegungen des Therapeuten erfasst, daraus die Trajektorie des Roboters berechnet und anschließend an ihn weiterleitet. Dabei ist es wünschenswert, dass der Roboter die Bewegung des Therapeuten vollständig mit seinen vorhandenen Gelenken nachbildet, d.h. der Roboter hat mindestens den gleichen Bewegungsraum wie der menschliche Arm und kann auf diese Weise seine Gelenke so steuern, dass am Ende die Bewegung des Armes komplett reproduziert wird. So wäre auf jeden Fall die Übungstrajektorie durch den Roboter exakt reproduzierbar, weil die Bahnplanung schon durch den Therapeuten erfolgt ist.

Als konzeptioneller Lösungsansatz für die oben genannten Punkte soll in dieser Arbeit eine Software entwickelt werden, die als Schnittstelle zwischen dem 3D Tracking-System und dem Armroboter fungiert.

Dabei wird zuerst die Bewegung des Therapeuten mit einem 3D-Tracking-System erfasst, dann die Information in einem Rechner verarbeitet, in eine geeignete Robotertrajektorie umgerechnet und an den Roboter übertragen. Somit kann die Bewegung auch bei Abwesenheit des Therapeuten beliebig oft wiederholt werden.

Die Software soll als Schnittstelle zu beiden beteiligten Systemen (3D-Tracking-System und Roboter) dienen. D.h. sie reguliert den Datenaustausch und kümmert sich um die Steuerung des Roboters. Ferner wird auch eine mögliche Abbildung der Bewegung des menschlichen Arms in eine Robotertrajektorie implementiert.

Kapitel 2

Benutzte Hardware und Software

Im Rahmen der Arbeit wurde mit dem Kuka LBR Knickarmroboter und dem Vicon Motion Capture System gearbeitet. In diesem Kapitel wird kurz auf diese zwei Systeme eingegangen.

2.1 Kuka LBR IV

Kuka LBR IV ist ein kompakter Leichtbauroboter, der vom DLR[1] entwickelt wurde. Dieser Roboter besitzt vier Rotationsgelenke sowie drei Torsionsgelenke, was die Möglichkeit bietet, die Bewegung eines menschlichen Armes annähernd nachzuahmen.

Das System besteht aus dem Roboter und einem Steuerrechner namens KRC[2], womit man die Robotereinstellungen vornehmen und Bewegungsfolgen programmieren kann. Die Programmierung wird mit einer extra für Kuka-Roboter entwickelten Programmiersprache, KRL[3], realisiert. Weil der Roboter nicht durch die KRL sondern durch das 3D-Tracking-System gesteuert werden soll, liegt der Schwerpunkt dieser Arbeit nicht in der KRL-Programmierung.

Zusätzlich zur lokalen Programmierung bietet das System eine Fernsteuerungsschnittstelle namens FRI[4] an, was eine Echtzeitsteuerung von einer externen Rechneranlage ermöglicht. Es handelt sich dabei um ein C++-Modul, das vom Benutzerpro-

[1] Deutsches Zentrum Luft- und Raumfahrt
[2] Kuka Robot Controller
[3] Kuka Robot Language
[4] Fast Research Interface

Abbildung 2.1: Kuka LBR IV mit dem Steuerrechner

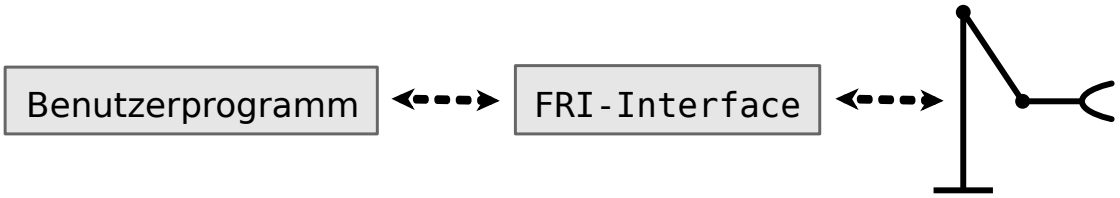

Abbildung 2.2: Das Kuka FRI-Interface

gramm aufgerufen und als Programmierschnittstelle benutzt wird (Abbildung 2.2). Die Schnittstelle besteht aus zwei Komponenten.

- ein KRL-Programm, was auf dem Kuka-Steuerrechner läuft. Es ist verantwortlich für die Initialisierung der Achssteifigkeiten, die Aktivierung des FRI-Interfaces und die Echtzeitkommunikation zwischen dem Roboter und dem Fernsteuerrechner.

- ein Programm, was auf dem Fernsteuerrechner läuft. Es ist verantwortlich für die Steuerung. Es empfängt die Istwerte vom Roboter und schickt nach anschließender Abarbeitung die Sollwerte zurück zum Roboter.

2.2 Vicon 3D-Tracking-System

Das Vicon 3D-Tracking-System Vicon MX besteht aus zehn Infrarotkameras (Vicon MX) für die Aufnahme und einer Rechneranlage (MX Giganet) sowie einer graphischen Oberfläche (Vicon Nexus). Das System ist selber in der Lage, die 3D-Trajektorien der verfolgten Objektpunkte aus den Kamerabildern zu berechnen und kann bis zu 500 Bilder pro Sekunde verarbeiten.

3D-Tracking von Objekten erfolgt durch kugelförmige Plastikmarker mit etwa 1 cm Durchmesser. Diese sind mit einer Folie umhüllt, die Infrarotlicht reflektiert und sind an zu verfolgenden Stellen aufgeklebt. Die Erkennung dieser Kugel erfolgt durch Infrarotkameras. Diese Kameras haben um das Objektiv Infrarot-LEDs, so dass die Marker einfach von anderen Objekten differenziert und registriert werden. Diese Kugeln werden von dem Vicon-System auch Marker genannt.

Die Positionsbestimmung beruht auf der Tatsache, dass beobachtete Objekte in der Kamera immer kleiner abgebildet werden, wenn sie sich entfernen. So lässt sich die Position eines Objektes bestimmen, wenn man das Objekt mit zwei Kameras aufnimmt, deren Brennpunkt und Abstand voneinander bekannt ist. In Abbildung 2.3 ist ein Testaufbau dargestellt, anhand dessen das Prinzip der Positionsbestimmung erläutert wird.

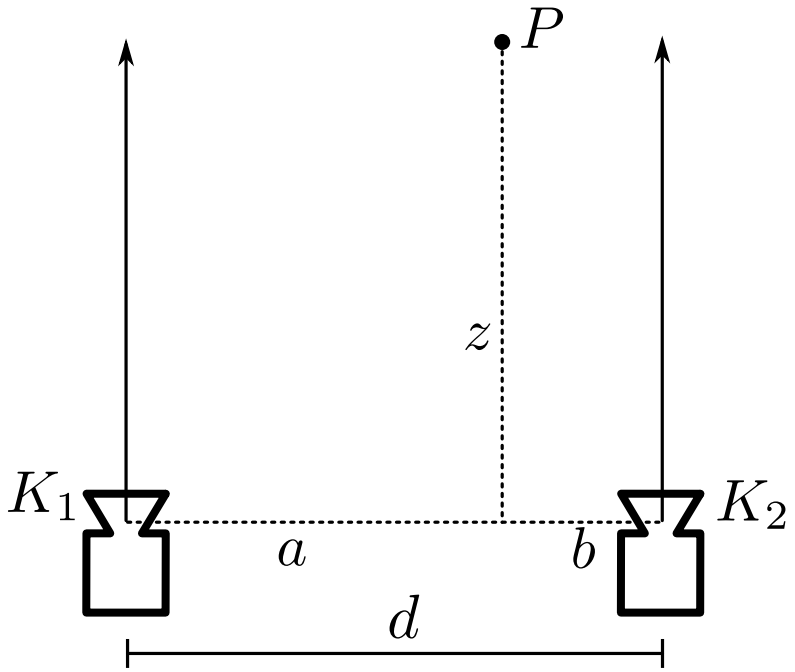

Abbildung 2.3: Einfacher Aufbau für die Berechnung der Objektposition

Im einfachen Testaufbau befinden sich zwei Kameras (K_1 und K_2), die auf der

gleichen Höhe wie der Objektpunkt P stehen und in die Pfeilrichtung gerichtet sind, so dass beide den Objektpunkt P sehen können. Der Abstand zwischen den Kameras ist fest eingestellt und beträgt $a + b = d$. Die Sensorposition beider Kameras liegt gleich weit entfernt von der Kameralinse und der Abstand beträgt f. Gesucht ist die Position vom Objektpunkt P.

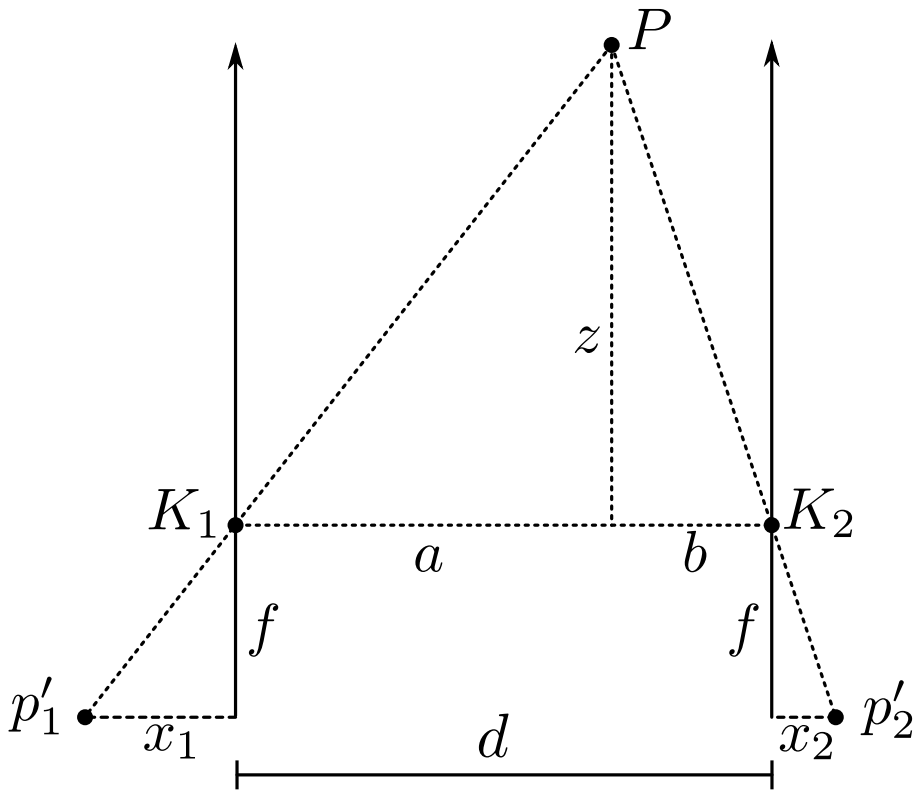

Abbildung 2.4: Hilfsdiagramm für die Herleitung der Objektposition

Wie in der Abbildung 2.4 dargestellt wird der Objektpunkt P auf den Sensoren auf die Punkte p'_1 sowie p'_2 abgebildet, die vom Sensorursprung jeweils x_1 und x_2 entfernt sind. Mit dem Ähnlichkeitssatz für die Dreiecke folgt:

$$\frac{a}{x_l} = \frac{z}{f} \quad \Rightarrow \quad a = \frac{x_l \cdot z}{f} \tag{2.1}$$

$$\frac{b}{x_r} = \frac{z}{f} \quad \Rightarrow \quad b = \frac{x_r \cdot z}{f} \tag{2.2}$$

Durch Addition der Gleichungen 2.1 und 2.2 lässt sich ein Ausdruck für den Abstand z ableiten (Gleichung 2.3).

$$a + b = (x_l + x_r) \cdot \frac{z}{f} \quad \Rightarrow \quad z = f(a+b) \cdot \frac{1}{x_l + x_r} \quad \Rightarrow \quad z = f \cdot d \cdot \frac{1}{x_l + x_r} \tag{2.3}$$

Mit bekanntem z lassen sich nun mit den Ähnlichkeitsgleichungen aus 2.1 und 2.2 die seitlichen Abstände a und b genauso berechnen. Wenn man den Term für z aus 2.3 einsetzt, folgt für a und b:

$$a = d \cdot \frac{x_l}{x_l + x_r} \tag{2.4}$$

$$b = d - a \tag{2.5}$$

Somit lässt sich die Objektposition mit a und z bzw. b und z bestimmen. Dieses Prinzip lässt sich auch auf den Fall übertragen, wenn der Blickpunkt der Kameras und der Objektpunkt nicht auf einer Ebene liegen oder wenn die Kameras horizontal gedreht sind.

Wie gezeigt, lässt sich die 3D-Position eines Objekts im Raum durch zwei Kameras bestimmen. Nichtsdestotrotz werden bei 3D-Tracking Systemen deutlich mehr als zwei Kameras benutzt. Durch diese Redundanz der Kameras werden einerseits Artefakte durch Nichtlinearitäten in den Kameralinsen vermieden und der mögliche Aufnahmeraum vergrößert. Andererseits werden auch die Verdeckungen von Markern reduziert, wenn man die Kameras um das verfolgende Objekt platziert.

Weil die 3D-Daten von Objekten online in einem externen PC verarbeitet werden, wird die Weiterleitungsfunktion von Vicon Nexus benutzt. Der weiterleitbare Datenstrom kann unter anderem unter Echtzeit erfolgen und wird durch den im Vicon Nexus eingebauten Vicon-Streaming-Server per TCP/IP Netzwerkprotokoll gestreamt.

Kapitel 3

Systemaufbau

3.1 Allgemeiner Systemaufbau

Das System besteht aus drei Komponenten. Zuerst werden die Armbewegungen mit dem 3D-Tracking-System registriert, dann werden die 3D-Daten in einem Computer verarbeitet und zuletzt wird der Armroboter mit den entsprechenden Gelenkpositionen angesteuert. Aus dieser Datenkette ergibt sich die folgende Struktur in Abbildung 3.1.

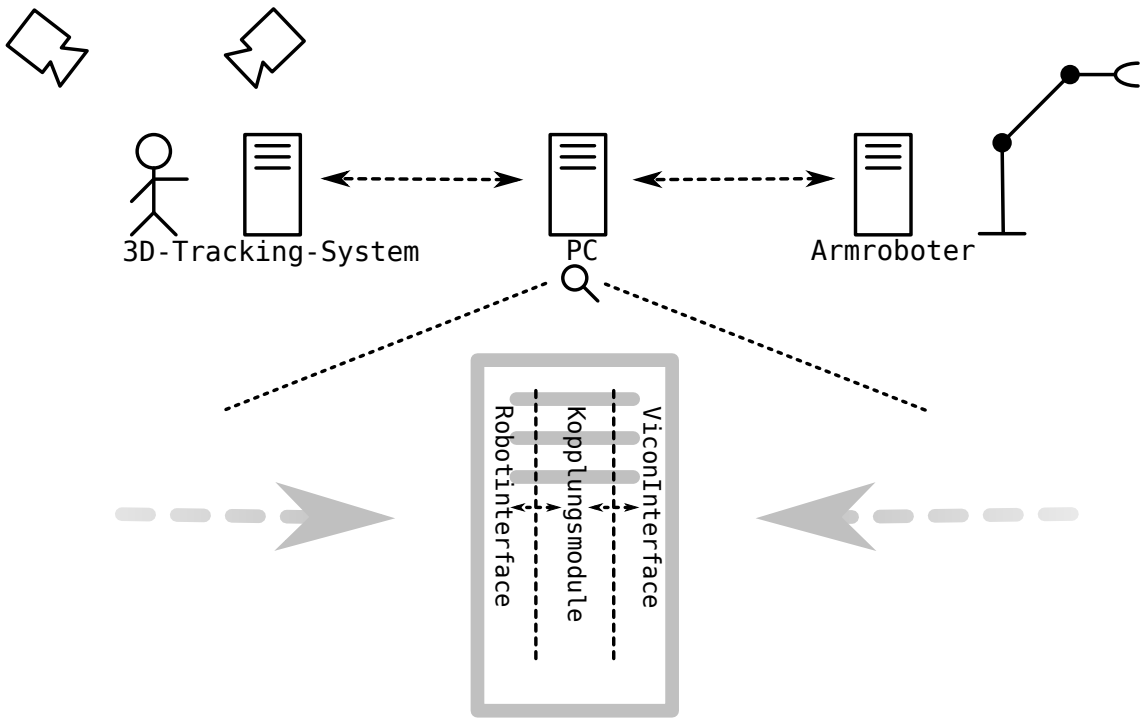

Abbildung 3.1: Allgemeiner Aufbau des Systems

Wie man der Abbildung 3.1 entnehmen kann, wird ein PC benutzt, um beide Systeme zu koppeln. Die Verarbeitungs-Software auf dem PC besteht im Wesentlichen aus diesen Komponenten:

- ViconInterface
- RobotInterface
- Kopplungsmodule

Das ViconInterface ist für die Initialisierung des Vicon-Streaming-Servers und den Empfang des Datenstroms vom Streaming-Server zuständig.

Das RobotInterface kümmert sich neben der Echtzeitverbindung zum Roboter und der Initialisierung auch um die Kontrolle und Umrechnung der Sollwerte, so dass sichergestellt wird, dass gültige Werte zum Roboter geschickt werden und somit ein Absturz der roboterinternen Software verhindert wird.

Die Kopplungsmodule sind in Abbildung 3.2 dargestellt. Der Parser ist für die Extraktion der 3D-Tracking-Daten, ViconFilter für die anschließende Filterung und ViconRobotBridge für die Berechnung der Roboter-Gelenkwinkeln sowie Regelung des Roboters zuständig. Sie werden anhand des Datenflusses vom 3D-Tracking-System bis zum Roboter im nächsten Unterkapitel ausführlich erläutert.

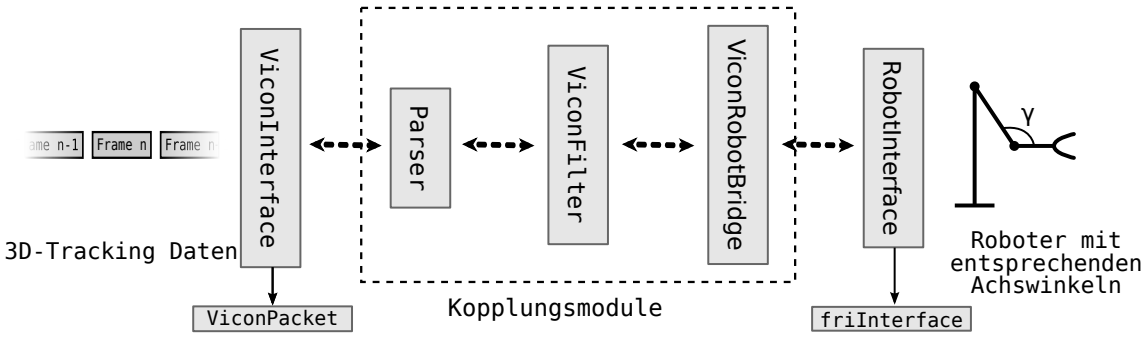

Abbildung 3.2: Klassenübersicht

3.2 Realisierung der Module

Die Kommunikation beginnt mit dem Empfang der 3D-Tracking Daten aus dem 3D-Tracking-System, die nacheinander in einer Klasse namens SocketBuffer gespeichert

werden. Für die Steuerung der Kommunikation wird dabei die im Modul Vicon-Packet definierte Datenstruktur benutzt. Weil aus dem Datenstrom zunächst nicht klar ist, wo sich welche Daten befinden, wird dieser Puffer zu dem Parser übergeben. Dieser Vorgang wird in der Abbildung 3.3 veranschaulicht.

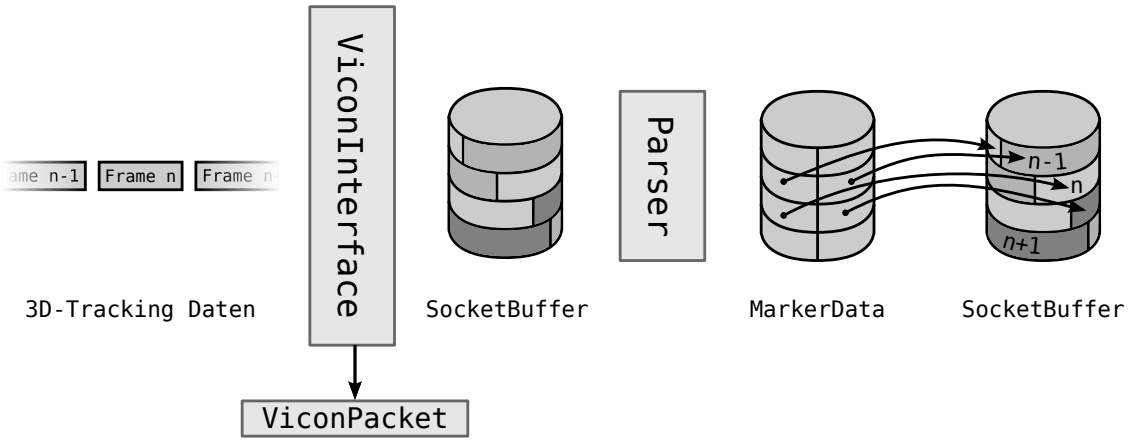

Abbildung 3.3: Datenflussdiagramm von 3D-Tracking-Daten bis zum Parser

Der Parser extrahiert die Daten aus den empfangenen Netzwerkpaketen. Dabei werden keine Daten aus SocketBuffer kopiert, sondern lediglich Zeiger auf entsprechende Daten in einem Container namens MarkerData gespeichert. Auf diese Weise bleibt SocketBuffer unberührt und dabei wird der Rechen- und Speicheraufwand durch Kopieren in einen anderen Container vermieden.

Bei diesen Daten handelt es sich um sogenannte Frames, die jeweils eine Bildsequenz darstellen. Ein Frame besteht aus einem Zeitstempel und 3D-Positionen von den Infrarot-Markern im Raum. MarkerData stellt diese Frames einem Verarbeitermodul in Form von Zeigern zur Verfügung. Die weitere Verarbeitung wird in der Abbildung 3.4 veranschaulicht.

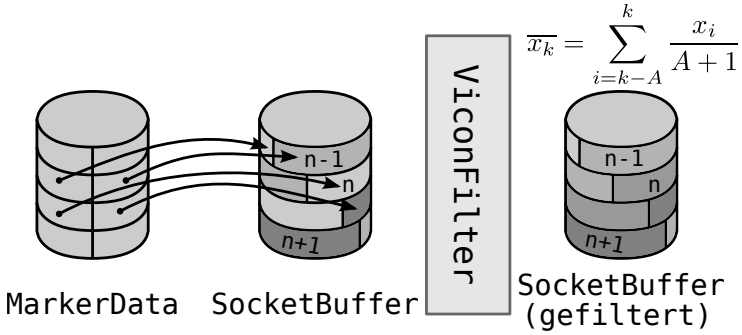

Abbildung 3.4: Datenflussdiagramm von ViconFilter

Weil alle eingesetzten Marker gleich aussehen, wird in Vicon Nexus vor dem eigentlichen Tracking der Marker durch den Benutzer benannt und im Datenstrom auch entsprechend gekennzeichnet, damit später das zu verarbeitende Modul weiss, zu welchem Gelenk das entsprechende 3D-Datum gehört. Die Unterscheidung und Zuordnung von Markern wird nach der Bennennung automatisch durch das 3D-Tracking-System durchgeführt und beruht auf dem ständigen Tracking der Marker. Solange sich die Marker im Aufnahmebereich der Kameras befinden, werden die Marker auch richtig zugeordnet, da die verwendete Aufnahme-Geschwindigkeit mit 200 Hz schnell genug ist, die gesamte Bewegung eines Markers zu verfolgen.

Wenn ein oder mehrere Marker aus dem Blickfeld der Kameras verschwinden und dann wieder eintauchen, muss das System meistens anhand der Struktur, die bei der Benennung von Markern eingegeben wurde, diese Marker unterscheiden und benennen. Diese Struktur wird durch die Verbindung von Markern mit Achsen und einer zusätzlichen Definition der Gelenke im Vicon Nexus weiter verbessert. Wenn zum Beispiel der Winkel des Ellenbogens bestimmt werden soll und dabei nur drei Marker am Schulter, Ellenbogen sowie Handgelenk benutzt werden, besteht die Gefahr, dass der Schulter- und Handgelenkmarker durch kurzzeitige Verdeckung vom Schultermarker falsch zugeordnet werden. Durch zusätzliche redundante Marker wird das Problem zum größten Teil vermieden.

Nichtsdestotrotz werden Marker ab und zu kurzzeitig falsch zugeordnet oder sind überhaupt nicht sichtbar. Damit diese verrauschten Daten nicht als Störgrößen an das Verarbeitermodul weitergeleitet werden, werden sie durch einen Tiefpassfilter herausgefiltert. Dieser ist in Abbildung 3.4 unter dem Namen ViconFilter zu finden.

Es handelt sich dabei um einen gleitenden Mittelwert-Filter, der nach der Formel 3.1 die Mittelwerte der letzten Markerpositionen berechnet und danach die letzte Position durch den berechneten Mittelwert ersetzt. Die Variable A bezeichnet dabei die Größe des gleitenden Mittelwert-Fensters. In dieser Arbeit wurde eine Fensterlänge von 10 gewählt.

$$\overline{x_k} = \sum_{i=k-A}^{k} \frac{a_i}{A+1} \tag{3.1}$$

Nachdem die Daten gefiltert wurden, wird aus den Markerdaten der entsprechende Roboterachswinkel berechnet. Diese Funktion wird hier als ViconRobotBridge bezeichnet. Die Verarbeitung wird in der Abbildung 3.5 veranschaulicht.

Um aus den Markerpositionen Roboterachswinkel zu berechnen, wird hier ein verein-

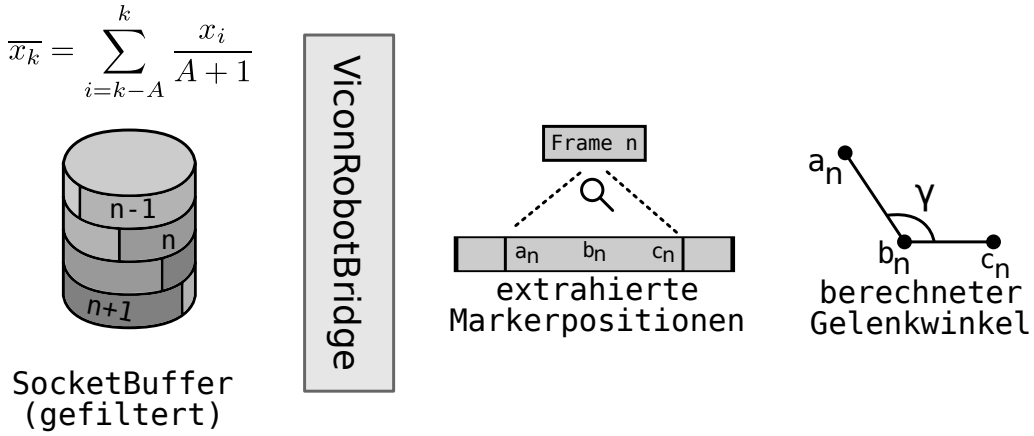

Abbildung 3.5: Datenflussdiagramm von ViconRobotBridge

fachtes Armmodell verwendet. Es besteht aus drei Gelenken; Schultergelenk, Ellenbogen und Handgelenk. Die Gelenke werden in dieser Arbeit alle als 1D-Scharniergelenke betrachtet. Sie passen anatomisch gut zu den Gelenken 2, 4 und 6 des Roboters, wie in Abbildung 3.6 gut zu erkennen ist.

Abbildung 3.6: Armroboter mit numerierten Achsen

Wie im Abschnitt 2.2 erläutert, werden die Marker benutzt, um Objektpunkte im Raum zu verfolgen. Die Marker werden wie in Abbildung 3.7 abgebildet auf die Schulter(s), den Ellenbogen(e), das Handgelenk(w) und die Hand(h) geklebt und die Vektoren $\vec{s}$, $\vec{e}$, $\vec{w}$, $\vec{h}$ repräsentieren die entsprechenden 3D-Positionen.

Um die Winkel der Armgelenke des Menschen zu bestimmen, werden die Koordina-

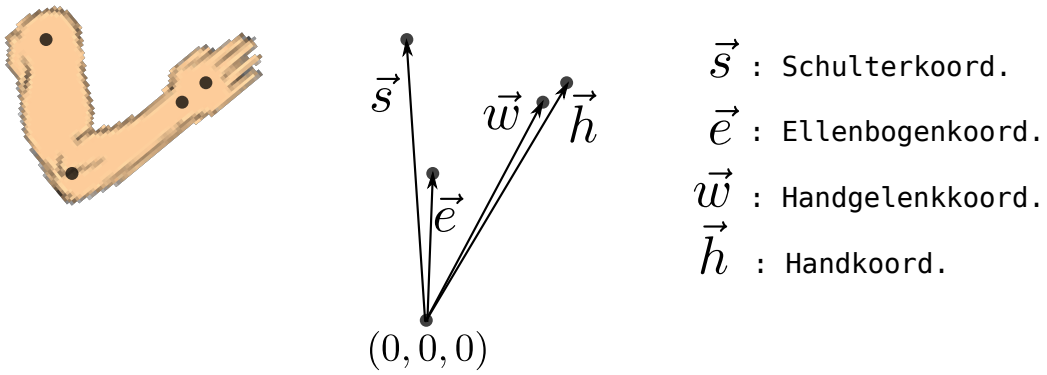

Abbildung 3.7: Die Markerpositionen und aus denen abgeleitete Vektoren

ten der Marker gelesen und daraus drei Vektoren berechnet. Diese Vektoren und die aus denen resultierenden Gelenkwinkel sind in der Abbildung 3.8 dargestellt.

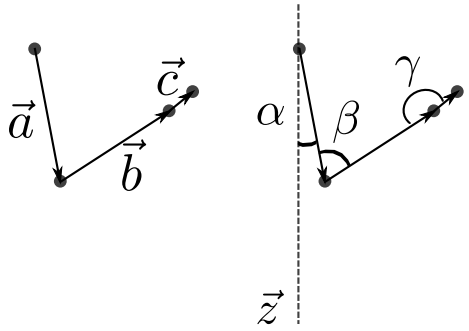

Abbildung 3.8: Armvektoren mit entsprechenden Winkeln

Die Vektoren $\vec{a}$ $\vec{b}$ und $\vec{c}$ resultieren aus den jeweiligen Differenzen der Markervektoren:

$$\vec{a} = \vec{e} - \vec{s} \quad \vec{b} = \vec{w} - \vec{e} \quad \vec{c} = \vec{h} - \vec{w} \tag{3.2}$$

Der Winkel zwischen zwei beliebigen Vektoren $\vec{x}$ und $\vec{y}$ lässt sich mit dem Skalarprodukt berechnen:

$$\vec{x} \cdot \vec{y} = |\vec{x}|\,|\vec{y}|\,\cos\angle(\vec{x},\vec{y}) \tag{3.3}$$

Somit lässt sich der Winkel für das Ellenbogengelenk β aus der Abbildung 3.8 mit

dieser Formel folgendermaßen berechnen:

$$\vec{a} \cdot \vec{b} = |\vec{a}|\,|\vec{b}|\,\cos(\beta) \tag{3.4}$$

Wenn diese Gleichung nach β aufgelöst wird, folgt:

$$\beta = \arccos\left(\frac{\vec{a} \cdot \vec{b}}{|\vec{a}|\,|\vec{b}|}\right) \tag{3.5}$$

Analog lassen sich die Winkel α für die Schulter und γ für das Handgelenk folgendermaßen berechnen:

$$\alpha = \arccos\left(\frac{\overrightarrow{(0\,0\,1)} \cdot \vec{a}}{|\vec{a}|}\right) \tag{3.6}$$

$$\gamma = \arccos\left(\frac{\vec{b} \cdot \vec{c}}{|\vec{b}|\,|\vec{c}|}\right) \tag{3.7}$$

Zur Steuerung des Roboters werden die Sollwinkel für die Gelenke in festen Zeitabschnitten von 20 ms zum Roboter geschickt. Dabei muss der Roboter in einem Zeitabschnitt diese Sollwerte anfahren. Wenn ein Gelenkmotor einen Sollwinkel nicht innerhalb dieses Zeitfensters anfahren kann, wird ein Fehler ausgegeben und der Roboter durch den Steuerrechner gestoppt.

Genauso müssen die Gelenkmotoren im Stoppzustand langsam angefahren werden, weil der Roboter selber eine Trägheit hat und die Servomotoren nur ein bestimmtes Drehmoment aufbringen können. Sonst wird der Roboter gestoppt.

Diese oben genannten zwei Punkte erfordern den Einsatz eines Reglers für die Sollwinkel. Dafür wurde in dieser Arbeit der folgende Regler benutzt:

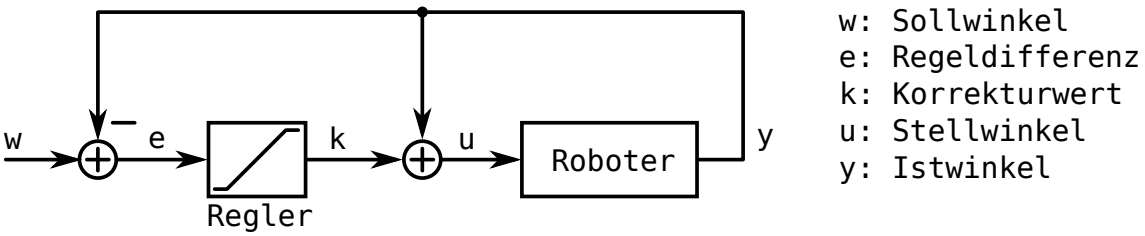

Abbildung 3.9: Winkelregelung

Die Sollwinkel werden zunächst mit den Istwinkeln verglichen und danach die Differenz an den Regler geschickt. Danach wird der Korrekturwert zum Istwinkel addiert, weil der Roboter nur absolute Winkelangaben akzeptiert.

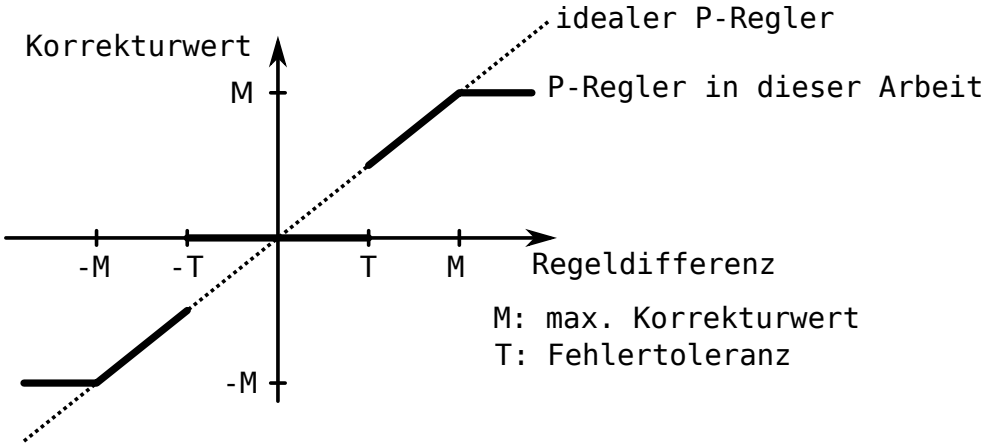

Abbildung 3.10: Regelungskennlinie des Reglers

Der Regler wird in der Abbildung 3.10 im Detail dargestellt. Er wurde in dieser Arbeit als Proportionalregler realisiert, der im oberen und unterem Bereich auf Grund der maximalen Achsgeschwindigkeit des Roboters nach oben und nach unten begrenzt ist. Zusätzlich gibt es auch den Toleranzbereich $[-T, T]$ für die Fehler, auf die nicht reagiert wird. Das vermeidet das unangenehme Ruckeln des Roboters, wenn der menschliche Arm in bewegungsloser Position bzw. die rekonstruierten Markerpositionen zittern.

Schließlich werden die Sollwerte mit der Hilfe von RobotInterface und friInterface zum Roboter geschickt.

Kapitel 4

Validierung

In diesem Kapitel soll überprüft werden, ob die Bewegung eines menschlichen Arms durch die in dieser Arbeit vorgestellte System und den Kuka LBR abgebildet werden kann.

Die Gelenke wurden in dieser Arbeit als 1D-Scharniergelenke betrachtet, deswegen können nicht alle Bewegungen abgebildet werden. Diese sind in der Tabelle 4.1 aufgelistet.

Gelenk	mögliche Bewegungen	fehlende Bewegungen
Schulter	Flexion, Extension	Abduktion, Adduktion, Innen- u. Außenrotation
Ellenbogen	Flexion, Extension	Pronation, Supination
Hand	Flexion, Extension	Abduktion, Adduktion

Tabelle 4.1: Roboterachsen und die entsprechenden Gelenke

Wie in Tabelle 4.1 ersichtlich sind in der Schulter zwei, in Ellenbogen und Hand jeweils ein Freiheitsgrad nicht abbildbar.

In der Abbildung 4.1 ist eine Bewegungssequenz dargestellt, welche die korrekte Wiedergabe der abbildbaren Gelenkbewegungen beim Einsatz des hier vorgestellten Systems zeigt. In den ersten zwei Bildern ist die Flexion und Extension des Ellenbogens, im zweiten und dritten Bild der Schulter sowie in den letzten zwei Bildern des Handgelenkes sichtbar.

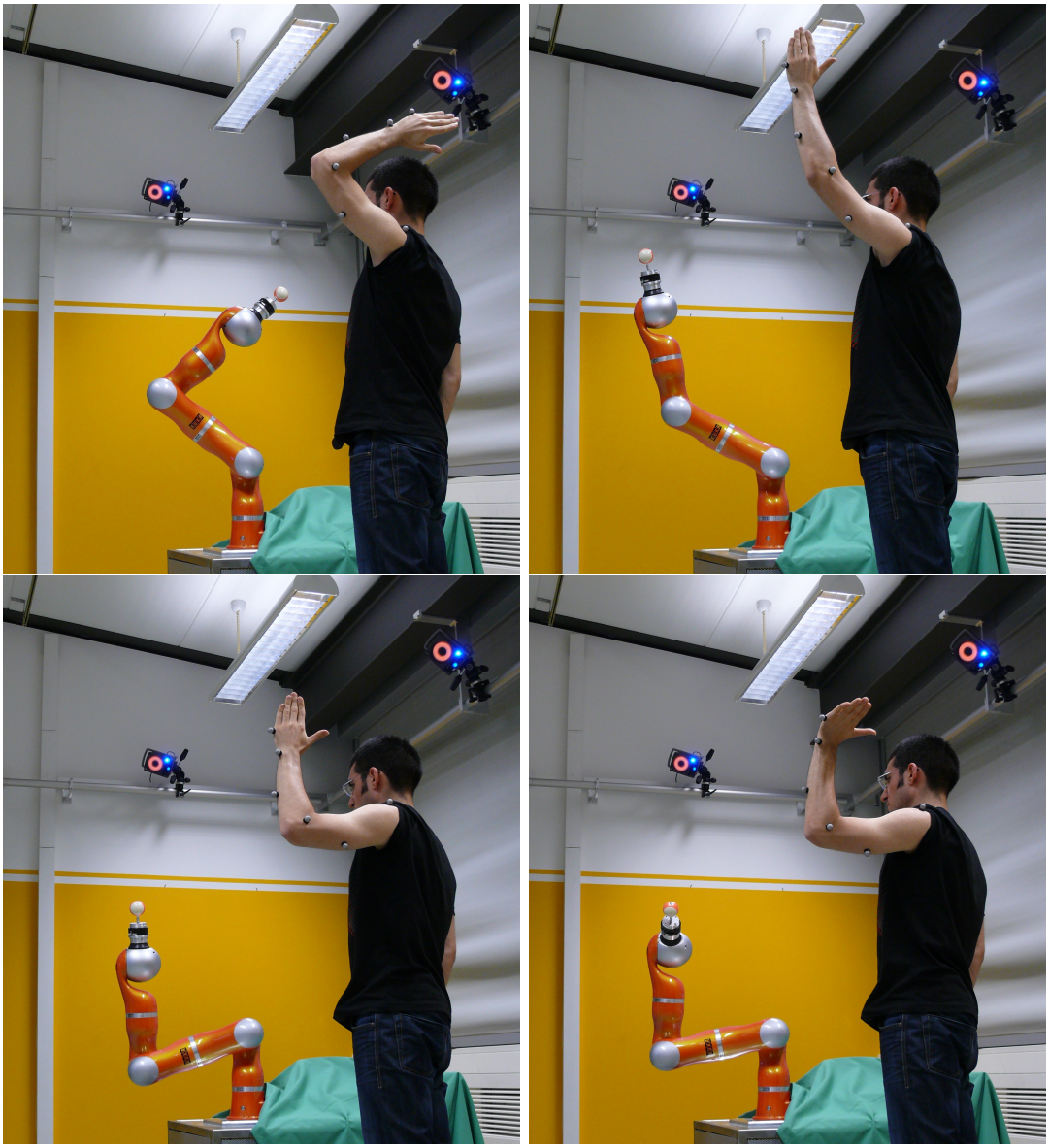

Abbildung 4.1: Bilder aus einer Bewegungssequenz

Kapitel 5

Zusammenfassung und Ausblick

Das Ziel der Arbeit war die Abbildung von menschlichen Armbewegungen auf einen Roboter. Es wurde eine Schnittstelle zwischen einem 3D-Tracking-System und einem Roboter sowie eine Abbildung aller Armgelenke als Scharniergelenke entwickelt.

Als nächster Schritt könnte der Schwerpunkt auf die anatomisch korrektere Abbildung der Armbewegungen auf dem Roboter gelegt werden, indem die Torsionsachsen des Roboters auch einbezogen werden. Ferner sollte das System zukünftig auch die Bewegungen in den bisher noch fehlenden Freiheitsgraden erfassen können.

Der Regler könnte auch verbessert werden, damit eine ruckelfreie Bewegung ermöglicht wird. Dazu gehört auch eine Geschwindigkeitsregelung, die eine identische Bewegung des Roboters mit der selben Geschwindigkeit wie der menschliche Arm ermöglicht.