



**FACHHOCHSCHUL - BACHELORSTUDIENGANG**

**Automatisierte Anlagen**

---

# **Berechnung der Inversen Kinematik für den Roboterarm des Rescue Roboters**

**als Bachelorarbeit eingereicht**

**zur Erlangung des akademischen Grades**

**Bachelor of Science in Engineering**

**VON**

**PENKNER Thomas**

**Juni 2011**

---

Betreuung der Bachelorarbeit durch

Prof. (FH) Dipl.-Ing. Dr. Burkhard Stadlmann



Campus **Wels**

Ich erkläre ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt, die den benutzten Quellen entnommenen Stellen als solche kenntlich gemacht habe und dass die Arbeit mit der vom Begutachter beurteilten Arbeit übereinstimmt.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt und auch nicht veröffentlicht.

.....  
Penkner Thomas

.....  
Engerwitzdorf, 16.06.2011

# Kurzfassung

Im Rahmen des Praktikums wurde die inverse Kinematik für den Roboterarm des Rescue Roboters berechnet. Dieser Roboterarm wurde speziell konstruiert, um den Anforderungen dieses Bewerbs gerecht zu werden. Der Kern dieser Arbeit beschäftigt sich daher damit, für vorgegebene Sollkoordinaten und Greiferorientierung die entsprechenden Achswinkel zu berechnen. Weiters beschreibt die Arbeit die Grundlagen der inversen Kinematik, sowie die Berechnung der Vorwärtskinematik. Es werden verschiedene Möglichkeiten zur Berechnung der inversen Kinematik aufgezeigt und kurz vorgestellt. Diese Arbeit beschäftigt sich hauptsächlich damit, den Aufgaben des Rescue Bewerbs handhaben zu können und beschreibt somit manche Teilgebiete mehr oder weniger detailliert. Es zeigt neben der Theorie lediglich eine von vielen Möglichkeiten zur Berechnung der inversen Kinematik für dieses System, wobei hierbei ein allgemeiner Lösungsweg mit iterativer Lösungsfindung gewählt wurde und somit nur Teile der Berechnung bei anderen Systemen angewendet werden können.

Desweiteren sollen die Ergebnisse dieser Arbeit als Grundlage für weitere Projekte mit Roboterarmen dienen.

# Abstract

As part of an bachelor project, the inverse kinematic for the rescue-robots robot arm was calculated. This robot arm was specifically designed to meet the requirements of the RobotRescue League. The paper describes the calculation of the angles of each axis with given position and orientation for the grapper. It also describes the basis knowledge of the inverse kinematic and the calculation of the forward kinematic. This work shows different possibilities for calculating the inverse kinematic and describes them. It mainly deals with finding a solution which meets the demands of the Rescue League, so some parts of this topic are explained more or less detailed. Beyond the theory it shows a possible way to calculate the inverse kinematic for this system, in which a general way with iterrativ solving system was used for finding a solution of the equations. So it is just possible to copy a few parts for different systems.

Furthermore the results of this work should also be useful for later projects with robotic arms.

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Motivation . . . . .	1
1.2	Der RoboRescue Bewerb . . . . .	1
<b>2</b>	<b>Grundlagen der inversen Kinematik</b>	<b>3</b>
2.1	Homogene Transformation . . . . .	3
2.2	Eklärung der inversen Kinematik . . . . .	5
2.3	Denavit-Hartenberg Parameter . . . . .	5
2.4	Jakobi Matrix . . . . .	6
<b>3</b>	<b>Berechnung des Systems „Roboterarm für den Rescue Roboter“</b>	<b>8</b>
3.1	Allgemein . . . . .	8
3.2	Berechnung der Vorwärtskinematik . . . . .	9
3.3	Berechnung der inversen Kinematik . . . . .	11
3.3.1	Allgemeiner Lösungsansatz . . . . .	11
3.3.2	Berechnung der Sollmatrix . . . . .	12
3.3.3	Zerlegung des Systems in Bereiche . . . . .	13
3.3.4	Lösen der einzelnen Gleichungssysteme . . . . .	14
3.3.5	Resultierender Fehler . . . . .	16
<b>4</b>	<b>Fazit und Dank</b>	<b>17</b>

# Kapitel 1

## Einleitung

### 1.1 Motivation

*„Es ist nicht genug zu wissen, man muss es auch anwenden;  
es ist nicht genug zu wollen, man muss es auch tun!“*

**Johann Wolfgang von Goethe, 1749 - 1832\***

Die Anforderungen an Roboter und automatisierte Anlagen steigen von Jahr zu Jahr, zudem reicht eine einfache automatisierte Datenaufnahme schon lange nicht mehr aus, um den Menschen zu entlasten. Roboter sollen heutzutage auch Gegenstände detektieren und hantieren können, um auch in diesem Bereich Aufgaben übernehmen zu können. Dies ist auch beim RoboCup Rescue Bewerb schon ein elementarer Bereich, um beste Platzierungen zu erreichen. Diese und weitere Gründe waren ausschlaggebend für den Start dieses Projektes.

### 1.2 Der RoboRescue Bewerb

Der Hauptbestandteil dieser Arbeit ist die Entwicklung und Berechnung eines Lösungsweges der inversen Kinematik für den Roboterarm des Rescue-Roboters des RoboRacing-Teams (RRT Wels<sup>1</sup>) der Fachhochschule in Wels. In der Zukunft sollen diese Roboter gefährliche Aufgaben, welche durch Brand, Feuer oder ähnliches hervorgerufen wurden, anstelle des Menschen verrichten. Gebaut und konstruiert wurde er, um sich beim RoboCup<sup>2</sup> Rescue Bewerb mit Teilnehmern der ganzen Welt zu messen. RoboCup ist eine Initiative zur Forschung in den Bereichen künstliche Intelligenz und autonome, mobile Roboter. Nähere Informationen hierzu sind auf der RoboCup-Webseite zu finden. Die Hauptaufgaben bei diesem Bewerb sind: Gebiet erkunden, eine 2D/3D Map erstellen, nach Lebenszeichen suchen und Objekte greifen und transportieren. Da die Anforderungen an die Roboter ständig größer werden und nun auch die Handhabung von Gegenständen ein elementarer Bestandteil des Bewerbs geworden ist, war eine Neuentwicklung des Roboterarms unumgänglich. *Abbildung 1.1* zeigt den Rescue Roboter der FH-Wels.

---

<sup>1</sup><http://rrt.fh-wels.at/>

<sup>2</sup><http://www.robocuprescue.org/>



Abbildung 1.1: Rescue Roboter MARK; oben: alt bzw. unten: neu[FW11]

Die Arena simuliert ein Katastrophengebiet und ist in mehrere Schwierigkeitsbereiche unterteilt, welche mit verschiedenen Farben gekennzeichnet werden. Sie basiert auf der Entwicklung der U.S. National Institute of Standards and Technology NIST<sup>3</sup>. [Wik11]



Abbildung 1.2: Rescue Arena [A.S08]

<sup>3</sup><http://www.nist.gov/index.html>

## Kapitel 2

# Grundlagen der inversen Kinematik

### 2.1 Homogene Transformation

In dieser Arbeit werden zur besseren Übersicht die zusammenhängenden Kinematiken des Roboterarms als „Objekte“ und deren in sich starren Elemente als „Sektionen“ bezeichnet. Derartige Sektionen haben jeweils ein körperfestes Koordinatensystem, welche definiert sind durch homogene Transformationsmatrizen (Frames), welche wiederum ein einziges festes Welt-Koordinatensystem in die Systeme der jeweiligen Sektionen überführen.

*„Zur Berechnung dieser Matrizen ist es notwendig zunächst Rotations- und Translationsbewegungen der einzelnen Sektionen zueinander darzustellen, also die relative Lage jeweils zweier benachbarter Sektionen zueinander in Abhängigkeit der jeweiligen variablen Achskoordinaten zu definieren. Das körperfeste Koordinatensystem einer Sektion  $n-1$  wird dabei durch Multiplikation mit einer von ihrer Achskoordinate abhängigen homogenen Transformationsmatrix in das körperfeste Koordinatensystem der ihr nachfolgenden Sektion  $n$  transformiert!“ [Sch98]*

Der grundsätzliche Aufbau einer Transformationsmatrix ist wie folgt

$$\underline{T}_{n-1,n} = \begin{bmatrix} u_x & v_x & w_x & p_x \\ u_y & v_y & w_y & p_y \\ u_z & v_z & w_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

wobei der Vektor  $\vec{p}$  den Positionsvektor und die Vektoren  $\vec{u}$ ,  $\vec{v}$ ,  $\vec{w}$  die Rotationsmatrix bilden. [Sta07]

Bei einem Translationsgelenk ist die Matrix definiert durch

$$\underline{T}_{n-1,n} = \begin{bmatrix} 1 & 0 & 0 & p_x \\ 0 & 1 & 0 & p_y \\ 0 & 0 & 1 & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

und beschreibt nur eine Verschiebung des Ursprungs. [Sta07]

Bei einem reinen Rotationsgelenk hat die homogene Transformationsmatrix die Form

$$\underline{T}_{n-1,n} = \begin{bmatrix} u_x & v_x & w_x & 0 \\ u_y & v_y & w_y & 0 \\ u_z & v_z & w_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

und beschreibt mit der 3x3 Orientierungsmatrix lediglich eine Raumdrehung. [Sta07]

Die Rotationen um die jeweiligen Achsen sind wie folgt definiert

$$Rot(x, \alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$Rot(y, \beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$$Rot(z, \gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 & 0 \\ \sin\gamma & \cos\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad [\text{Sta07}]$$

Im folgenden gelten zwecks einfacher Notation die Abkürzungen

$$\underline{Pos}(\underline{T}) = [p_x, p_y, p_z]^T$$

$$\underline{Rot}(\underline{T}) = \begin{bmatrix} u_x & v_x & w_x \\ u_y & v_y & w_y \\ u_z & v_z & w_z \end{bmatrix} = [\underline{Rot}_0(\underline{T}), \underline{Rot}_1(\underline{T}), \underline{Rot}_2(\underline{T})]$$

Sämtliche sektionsfeste Koordinatensysteme entstehen aus dem Basis Koordinatensystem des Objektes multipliziert mit sämtlichen homogenen Transformationsmatrizen der kinematischen Kette bis zu der betrachteten Sektion. Die ersten drei Zeilen dieses Matrizenproduktes definieren also die aus Position und Orientierung bestehende Lage einer Sektion im Raum. Das Basis-Koordinatensystem eines Objektes ergibt sich analog durch Multiplikation des Welt-Koordinatensystems mit der homogenen Transformationsmatrix der Objektbasis (Vorwärtskinematik). [Ang07]

$$\underline{T}_{Welt,3} = \underline{T}_{Welt,Basis} \cdot \underline{T}_{Basis,1} \cdot \underline{T}_{1,2} \cdot \underline{T}_{2,3}$$

## 2.2 Erklärung der inversen Kinematik

Im Gegensatz zur in *Kapitel 2.1* vorgestellten Vorwärtskinematik, wo ein Bezug des Welt- oder auch eines Basiskoordinatensystems zu einem Endeffektor hergestellt wird, beschreibt die inverse Kinematik den Zusammenhang zwischen dem Endpunkt bzw. Endeffektor und einem Basis- oder Weltkoordinatensystem. Diese Vorgehensweise ist sehr nützlich, da in der Realität meist nur die gewünschte Position des Endeffektors bekannt ist. Durch bestimmte Beschränkungen der Gelenke und Achsen ist es möglich, die Bewegungen zu kontrollieren und Kollisionen oder unnötig lange Verfahrwege zu vermeiden. Spezielle Algorithmen optimieren diese Dinge je nach Anforderung noch weiter. Für die Berechnung der inversen Kinematik ist es meist notwendig zuerst die Vorwärtskinematik des Modells zu bestimmen. Es gilt für das inverse System mit dem Weltkoordinatensystem  $\underline{T}_{Welt}$  und dem Endeffektorkoordinatensystem  $\underline{T}_{Welt,3}$  folgender Zusammenhang

$$\underline{T}_{3,Welt} = [\underline{T}_{Welt,Basis} \cdot \underline{T}_{Basis,1} \cdot \underline{T}_{1,2} \cdot \underline{T}_{2,3}]^{-1} = \underline{T}_{Welt,3}^{-1} \quad [\text{Ang07}]$$

## 2.3 Denavit-Hartenberg Parameter

Da die Beziehung zwischen zwei Sektions-Koordinatensystemen nur in einigen Fällen eine reine Translation oder eine reine Rotation darstellt, ist eine allgemeinere Transformationsvorschrift notwendig, die auch beliebig zueinander verlaufende Gelenkachsen berücksichtigt. Eine solche allgemeine Beschreibung liefert die Parametrisierung nach Denavit-Hartenberg (*Abbildung 2.1*). Danach wird die relativ komplizierte Transformation des Koordinatensystems  $n-1$  in das Koordinatensystem  $n$  durch folgende vier nacheinander auszuführenden Elementartransformationen beschrieben:

$$T_{n-1,n} = \underline{Rot}(z_{n-1}, \Theta_n) \cdot \underline{Trans}(0, 0, d_n)^T \cdot \underline{Trans}(a_n, 0, 0)^T \cdot \underline{Rot}(x_n, \alpha_n) \quad [\text{Ang07}]$$

Hierbei ist  $\underline{Rot}(v, \alpha)$  eine einfache Rotation um den Vektor  $v$  mit dem Winkel  $\alpha$  und  $\underline{Trans}(x, y, z)^T$  eine translatorische Bewegung entlang des Vektors  $[x, y, z]^T$ . [Sta07]

Erläuterung der Denavit-Hartenberg Parameter:

$\Theta_n$  = Winkel zwischen Achse  $x_{n-1}$  und Achse  $x_n$

$a_n$  = Länge der einzigen gemeinsamen Normalen der Gelenkachsen

$d_n$  = Länge vom Ursprung des Koordinatensystems  $n-1$  bis zum Schnittpunkt der gemeinsamen Normalen mit der Achse  $z_{n-1}$

$\alpha_n$  = Winkel zwischen Achse  $z_{n-1}$  und der Achse  $z_n$  nach der Drehung um die  $x_n$  Achse



$$\underline{J}_n = \begin{bmatrix} z_{n-1} \\ \underline{0} \end{bmatrix}. [\text{Ang07}]$$

Für ein rotatorisches Gelenk gilt

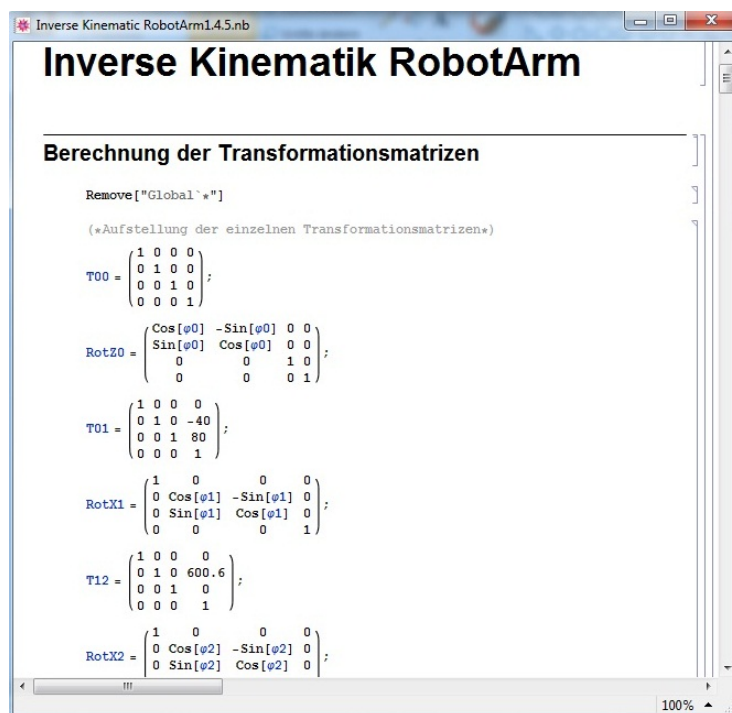
$$J_n = \begin{bmatrix} z_{n-1} \times r_{n-1} \\ z_{n-1} \end{bmatrix}. [\text{Ang07}]$$

## Kapitel 3

# Berechnung des Systems „Roboterarm für den Rescue Roboter“

### 3.1 Allgemein

Die Bestimmung mehrachsiger Roboterarmsysteme kann sehr schnell zu einer aufwendigen und komplexen Rechnung werden. Für die Berechnung des mathematischen Modells wurde daher das Computer Algebra Programm *MATHEMATICA* der Firma Wolfram verwendet. Dieses Programm kann numerische, sowie auch symbolische Aufgaben lösen, welche vom *MATHEMATICA* Kernel berechnet und in einem Notebook File (*Abbildung 3.1*) ausgegeben werden. Später wurde dann versucht die allgemeinen Ergebnisse zu vereinfachen und aufzuteilen, um eine Lösung der Gleichungssysteme mit einfacheren Mitteln zu ermöglichen.



```
Inverse Kinematik RobotArm1.4.5.nb

Inverse Kinematik RobotArm

Berechnung der Transformationsmatrizen

Remove["Global`*"]

(*Aufstellung der einzelnen Transformationsmatrizen*)

T00 =  $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$ 

RotZ0 =  $\begin{pmatrix} \text{Cos}[\varphi_0] & -\text{Sin}[\varphi_0] & 0 & 0 \\ \text{Sin}[\varphi_0] & \text{Cos}[\varphi_0] & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$ 

T01 =  $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -40 \\ 0 & 0 & 1 & 80 \\ 0 & 0 & 0 & 1 \end{pmatrix};$ 

RotX1 =  $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \text{Cos}[\varphi_1] & -\text{Sin}[\varphi_1] & 0 \\ 0 & \text{Sin}[\varphi_1] & \text{Cos}[\varphi_1] & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$ 

T12 =  $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 600.6 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$ 

RotX2 =  $\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \text{Cos}[\varphi_2] & -\text{Sin}[\varphi_2] & 0 \\ 0 & \text{Sin}[\varphi_2] & \text{Cos}[\varphi_2] & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix};$ 
```

Abbildung 3.1: Screenshot Mathematica Notebookfile

Da die Anforderungen nur eine Position mit Orientierung des Greifpunktes und keine vordefinierte Geschwindigkeit verlangen, wurde (wie sehr oft auch in der Industrie) auch in diesem Fall auf die Anwendung bekannter Systeme, wie beispielsweise die Jakobi Matrix oder das Verfahren nach Denavit Hartenberg (*Kapitel 2.3* und *2.4*) verzichtet und zur Vereinfachung ein allgemeiner Lösungsweg gewählt. Dieser Ansatz wird sehr oft gewählt, da so (der Achsanordnung angepasst) das System beispielsweise in einzelne lösbare Bereiche aufgeteilt oder möglicherweise sogar allgemein vereinfacht werden kann.

### 3.2 Berechnung der Vorwärtskinematik

Um die Vorwärtskinematik zu bestimmen, werden zunächst für das System „Roboterarm für den Rescue Roboter“ (*Abbildung 3.2*) die einzelnen Transformationsmatrizen aufgestellt. Das Worldframe wurde in diesem Fall im Montagepunkt ( $T_0$ ) des Roboterarms definiert und kann über die Achse  $L_0$  rotatorisch um die z-Achse manipuliert werden. Somit ergibt sich für das erste Basiskoordinatensystem die Transformationsmatrix

$$\underline{T}_{World,0} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

mit einer relativen Rotation

$$\underline{Rot}_{z,0} = \begin{bmatrix} \cos(\varphi_0) & -\sin(\varphi_0) & 0 & 0 \\ \sin(\varphi_0) & \cos(\varphi_0) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

Weiters ergeben sich gleichermaßen aus den mechanischen Gegebenheiten für die kinematische Kette des Systems folgende Transformationsmatrizen

$$\underline{T}_{0,1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -40 \\ 0 & 0 & 1 & 80 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{Rot}_{x,1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\varphi_1) & -\sin(\varphi_1) & 0 \\ 0 & \sin(\varphi_1) & \cos(\varphi_1) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{T}_{1,2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 600,6 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{Rot}_{x,2} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\varphi_2) & -\sin(\varphi_2) & 0 \\ 0 & \sin(\varphi_2) & \cos(\varphi_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{T}_{2,3} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 450,1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{Rot}_{x,3} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\varphi_3) & -\sin(\varphi_3) & 0 \\ 0 & \sin(\varphi_3) & \cos(\varphi_3) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{T}_{3,4} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 470,3 + L \\ 0 & 0 & 1 & 59,5 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{Rot}_{y,4} = \begin{bmatrix} \cos(\varphi_4) & 0 & \sin(\varphi_4) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\varphi_4) & 0 & \cos(\varphi_4) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{Rot}_{x,5} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\varphi_5) & -\sin(\varphi_5) & 0 \\ 0 & \sin(\varphi_5) & \cos(\varphi_5) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\underline{T}_{4,TCP} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 39 + TCP \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

wobei  $\varphi_n$  die einzelnen Winkel der Achsen,  $L$  die Stellung des Linearantriebs und  $TCP$  die Greiferparameter definieren.

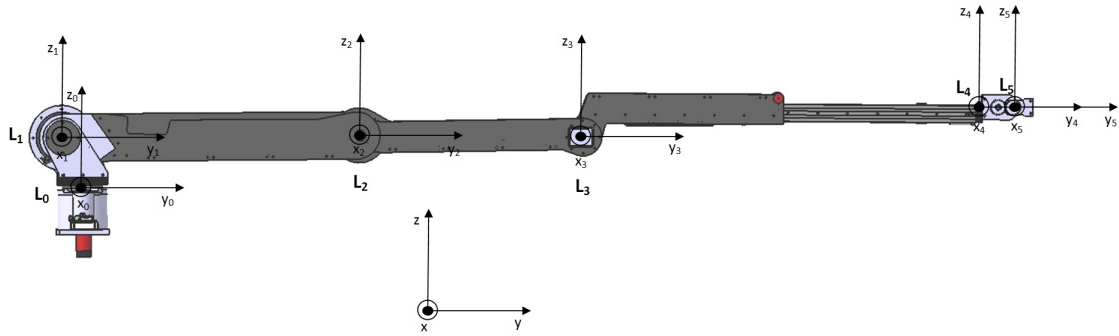


Abbildung 3.2: Anordnung und Orientierung der Frames im System

Auf die genaue Funktionsweise und Abmaße der einzelnen Sektionen und Objekte wird an dieser Stelle nicht weiter eingegangen, sondern auf die Literatur und Bachelorarbeit „Konstruktion eines Roboterarms für den Rescue Roboter“ verwiesen.

Nach Gleichung 2.1 lässt sich nun für die Beziehung zwischen Worldframe und TCP (Tool Center Point) die Transformationsmatrix  $\underline{T}_{World,TCP}$  durch einfache Multiplikation der einzelnen Transformations- und Rotationsmatrizen berechnen (Abbildung 3.3). [PBS08]

$$\underline{T}_{World,TCP} = \underline{T}_{World,0} \cdot \underline{T}_{0,1} \cdot \underline{T}_{1,2} \cdot \underline{T}_{2,3} \cdot \underline{T}_{3,4} \cdot \underline{T}_{4,TCP}$$

```

Out[134]= {{Cos[φ0] Cos[φ4] - (Cos[φ3] (Cos[φ2] Sin[φ0] Sin[φ1] + Cos[φ1] Sin[φ0] Sin[φ2]) - (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) Sin[φ4],
Cos[φ5] (Cos[φ3] (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) + (Cos[φ2] Sin[φ0] Sin[φ1] - Cos[φ1] Sin[φ0] Sin[φ2]) Sin[φ3]) +
(Cos[φ4] (Cos[φ3] (Cos[φ2] Sin[φ0] Sin[φ1] + Cos[φ1] Sin[φ0] Sin[φ2]) - (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) + Cos[φ0] Sin[φ4]) Sin[φ5],
Cos[φ5] (Cos[φ4] (Cos[φ3] (Cos[φ2] Sin[φ0] Sin[φ1] + Cos[φ1] Sin[φ0] Sin[φ2]) - (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) - Cos[φ0] Sin[φ4]) -
(Cos[φ3] (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) + (Cos[φ2] Sin[φ0] Sin[φ1] - Cos[φ1] Sin[φ0] Sin[φ2]) Sin[φ3]) Sin[φ5],
40 Sin[φ0] - 600.6 Cos[φ1] Sin[φ0] + 450.1 (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) +
(470.3 + L) (Cos[φ3] (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) - (Cos[φ2] Sin[φ0] Sin[φ1] - Cos[φ1] Sin[φ0] Sin[φ2]) Sin[φ3]) +
59.5 (Cos[φ3] (Cos[φ2] Sin[φ0] Sin[φ1] + Cos[φ1] Sin[φ0] Sin[φ2]) - (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) +
(39 - TCP) (Cos[φ5] (Cos[φ3] (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) + (Cos[φ2] Sin[φ0] Sin[φ1] - Cos[φ1] Sin[φ0] Sin[φ2]) Sin[φ3]) +
(Cos[φ4] (Cos[φ3] (Cos[φ2] Sin[φ0] Sin[φ1] + Cos[φ1] Sin[φ0] Sin[φ2]) - (-Cos[φ1] Cos[φ2] Sin[φ0] + Sin[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) - Cos[φ0] Sin[φ4]) Sin[φ5]),
(Cos[φ4] Sin[φ0] - (Cos[φ3] (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) - (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) + Sin[φ0] Sin[φ4]) Sin[φ5],
Cos[φ5] (Cos[φ3] (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) + (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) Sin[φ3]) +
(Cos[φ4] (Cos[φ3] (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) - (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) + Sin[φ0] Sin[φ4]) Sin[φ5],
Cos[φ5] (Cos[φ4] (Cos[φ3] (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) - (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) + Sin[φ0] Sin[φ4]) -
(Cos[φ3] (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) + (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) Sin[φ3]) Sin[φ5],
-40 Cos[φ0] + 600.6 Cos[φ0] Cos[φ1] + 450.1 (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) +
(470.3 + L) (Cos[φ3] (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) + (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) Sin[φ3]) +
59.5 (Cos[φ3] (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) - (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) +
(39 - TCP) (Cos[φ5] (Cos[φ3] (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) + (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) Sin[φ3]) +
(Cos[φ4] (Cos[φ3] (-Cos[φ0] Cos[φ2] Sin[φ1] - Cos[φ0] Cos[φ1] Sin[φ2]) - (Cos[φ0] Cos[φ1] Cos[φ2] - Cos[φ0] Sin[φ1] Sin[φ2]) Sin[φ3]) + Sin[φ0] Sin[φ4]) Sin[φ5]),
(-Cos[φ3] (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) - (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) Sin[φ3]) Sin[φ4],
Cos[φ5] (Cos[φ3] (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) + (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) Sin[φ3]) +
Cos[φ4] (Cos[φ3] (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) - (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) Sin[φ3]) Sin[φ5],
Cos[φ4] Cos[φ5] (Cos[φ3] (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) - (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) Sin[φ3]) -
(Cos[φ3] (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) + (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) Sin[φ3]) Sin[φ5],
80 + 600.6 Sin[φ1] + 450.1 (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) + 59.5 (Cos[φ3] (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) - (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) Sin[φ3]) +
(470.3 + L) (Cos[φ3] (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) + (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) Sin[φ3]) -
(39 - TCP) (Cos[φ5] (Cos[φ3] (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) + (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) Sin[φ3]) +
Cos[φ4] (Cos[φ3] (Cos[φ1] Cos[φ2] - Sin[φ1] Sin[φ2]) - (Cos[φ2] Sin[φ1] + Cos[φ1] Sin[φ2]) Sin[φ3]) Sin[φ5]), {0, 0, 0, 1}}
    
```

Abbildung 3.3: Berechnung der Vorwärtskinematik mit Mathematica

### 3.3 Berechnung der inversen Kinematik

#### 3.3.1 Allgemeiner Lösungsansatz

Für die Berechnung der inversen Kinematik soll die absolute Position des TCP 's und dessen Orientierung vorgegeben werden und somit als Berechnungsgrundlage dienen. Mit Hilfe dieser Werte

wurden die Sollwerte für die Matrix  $\underline{T}_{World,TCP}$  berechnet (siehe *Kapitel 3.3.2*). Zur Vereinfachung der Lösungsfindung der Gleichungssysteme wurde anschließend das System in Sektionen zerlegt und versucht so weit wie möglich zu vereinfachen (siehe *Kapitel 3.3.3*). Durch Lösen der Gleichungssysteme (iterative Näherungsverfahren) konnten danach die einzelnen Winkel und Positionen bestimmt werden (*Kapitel 3.3.4*).

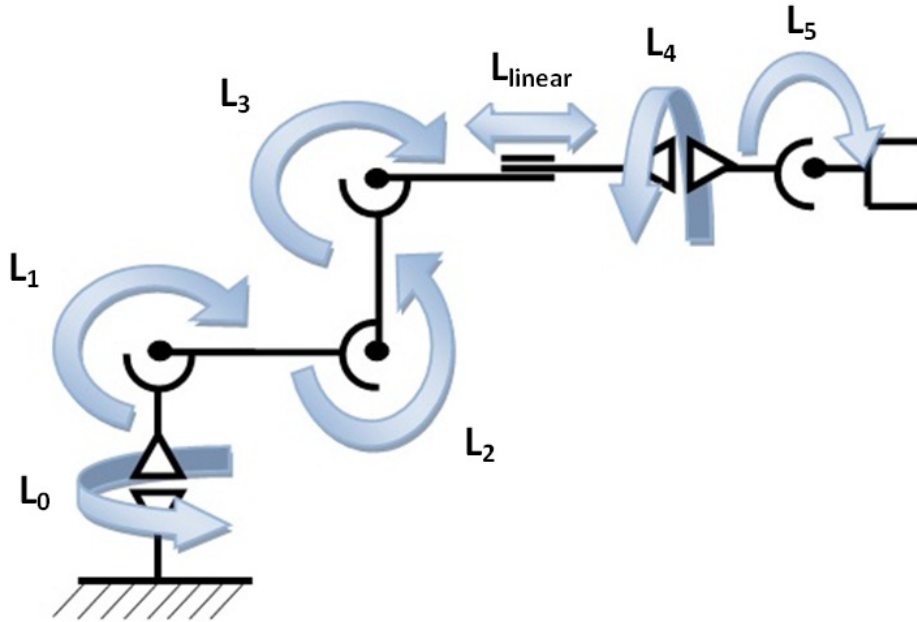


Abbildung 3.4: Schematische Darstellung des Roboterarms

### 3.3.2 Berechnung der Sollmatrix

Durch die Vorgabe der Position und Orientierung des TCP's ergibt sich eine Sollmatrix für  $\underline{T}_{World,TCP}$  von

$$\underline{T}_{World,TCPsoll} = \begin{bmatrix} 1 & 0 & 0 & P_{Xsoll} \\ 0 & 1 & 0 & P_{Ysoll} \\ 0 & 0 & 1 & P_{Zsoll} \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\varphi_{Xsoll}) & -\sin(\varphi_{Xsoll}) & 0 \\ 0 & \sin(\varphi_{Xsoll}) & \cos(\varphi_{Xsoll}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\varphi_{Ysoll}) & 0 & \sin(\varphi_{Ysoll}) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\varphi_{Ysoll}) & 0 & \cos(\varphi_{Ysoll}) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos(\varphi_{Zsoll}) & -\sin(\varphi_{Zsoll}) & 0 & 0 \\ \sin(\varphi_{Zsoll}) & \cos(\varphi_{Zsoll}) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

welche aus einer translatorischen Verschiebung und drei relativen, rotatorischen Drehungen (drei Achsen im Raum) besteht. Diese Sollwerte werden für jede Punkt- und Orientierungsvorgabe berechnet und sind die Grundlage für das Lösen der Gleichungssysteme, da nun die allgemein berechnete Vorwärtskinematik (*Kapitel 3.3*) mit den Sollwerten verglichen werden kann und somit insgesamt 12 lösbare Gleichungen entstehen.

### 3.3.3 Zerlegung des Systems in Bereiche

Das gesamte kinematische System wurde in drei Bereiche unterteilt (*Abbildung 3.5*), um die Lösung des Gleichungssystems zu vereinfachen.

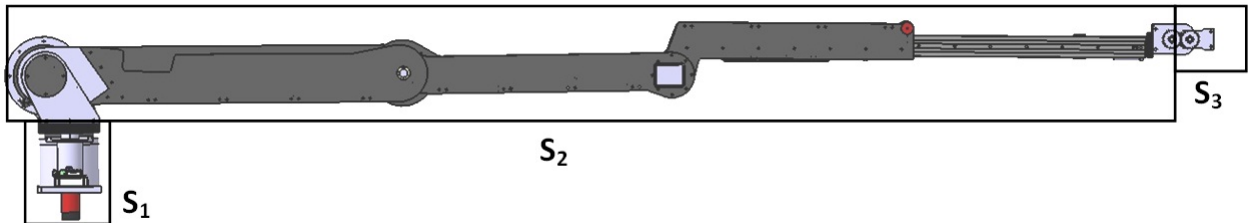


Abbildung 3.5: Einteilung des Systems in Sektionen

Da die Sollposition und Orientierung des Greifers fix vorgegeben wird, besitzt aufgrund der Anordnung der Achsen auch das Koordinatensystem  $L_5$  eine feste Position und Orientierung im Raum, auf welche zurückgerechnet werden kann (*Gleichung 3.3.3*). Das System wurde so auf die Bereiche  $S_1$  und  $S_2$  reduziert (*Abbildung 3.6*).

$$\underline{T}_{World,4soll} = \underline{T}_{World,TCPsoll} \cdot \underline{T}_{4,TCP}^{-1} = \begin{bmatrix} u_{x4soll} & v_{x4soll} & w_{x4soll} & P_{x4soll} \\ u_{y4soll} & v_{y4soll} & w_{y4soll} & P_{y4soll} \\ u_{z4soll} & v_{z4soll} & w_{z4soll} & P_{z4soll} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Da in den Abschnitten  $S_1$  und  $S_2$  nur die erste Achse eine Manipulation um die z-Achse zulässt, ist diese somit fix definiert und der Winkel kann ganz einfach über die Formel

$$\varphi_0 = \arctan\left(\frac{P_{x4soll}}{P_{y4soll}}\right)$$

berechnet werden. Der Bereich  $S_2$  wurde somit isoliert und kann gesondert berechnet werden.

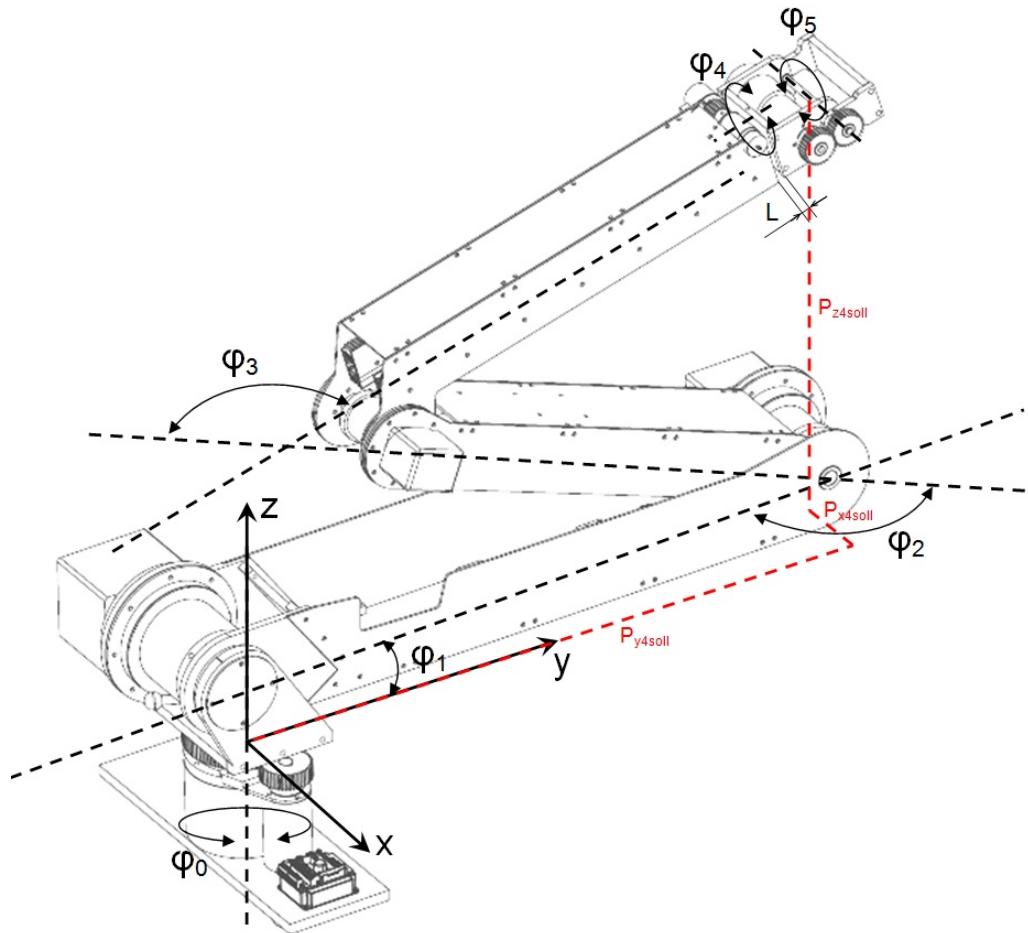


Abbildung 3.6: Reduziertes System

### 3.3.4 Lösen der einzelnen Gleichungssysteme

Da der Bereich  $S_2$  nur zweidimensionale Bewegungen zulässt, wird zur Berechnung der Gleichungssysteme über den Winkel  $\varphi_0$  der gesamte Roboterarm in die Ebene  $y/z$  gedreht. Die Positionsvorgabe  $P_{z4soll}$  ändert sich dabei nicht, jedoch muss  $P_{y4soll}$  umgerechnet werden (Abbildung 3.7). Es gilt

$$P_{y4soll2D} = \sqrt{P_{x4soll}^2 + P_{y4soll}^2}$$

$$P_{z4soll2D} = P_{z4soll}$$

Es wurde auf diese Weise auf ein zweidimensionales System mit vier Variablen reduziert, für welches allerdings wieder erneut die Vorwärtskinematik berechnet werden muss. Dies geschieht an dieser Stelle in vektorieller Form.

$$\underline{T}_{World,0/2D} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\underline{T}_{0,1/2D} = \begin{bmatrix} -40 \\ 80 \end{bmatrix}$$

$$\underline{T}_{1,2/2D} = \begin{bmatrix} \cos(\varphi_1) \cdot 600,6 \\ \sin(\varphi_1) \cdot 600,6 \end{bmatrix}$$

$$\underline{T}_{2,3/2D} = \begin{bmatrix} \cos(\varphi_1 + \varphi_2) \cdot 450,1 \\ \sin(\varphi_1 + \varphi_2) \cdot 450,1 \end{bmatrix}$$

$$\underline{T}_{3,4/2D} = \begin{bmatrix} \cos(\varphi_1 + \varphi_2 + \varphi_3) \cdot (470,3 + L) + \sin(\varphi_1 + \varphi_2 + \varphi_3) \cdot 59,9 \\ \sin(\varphi_1 + \varphi_2 + \varphi_3) \cdot (470,3 + L) + \cos(\varphi_1 + \varphi_2 + \varphi_3) \cdot 59,9 \end{bmatrix}$$

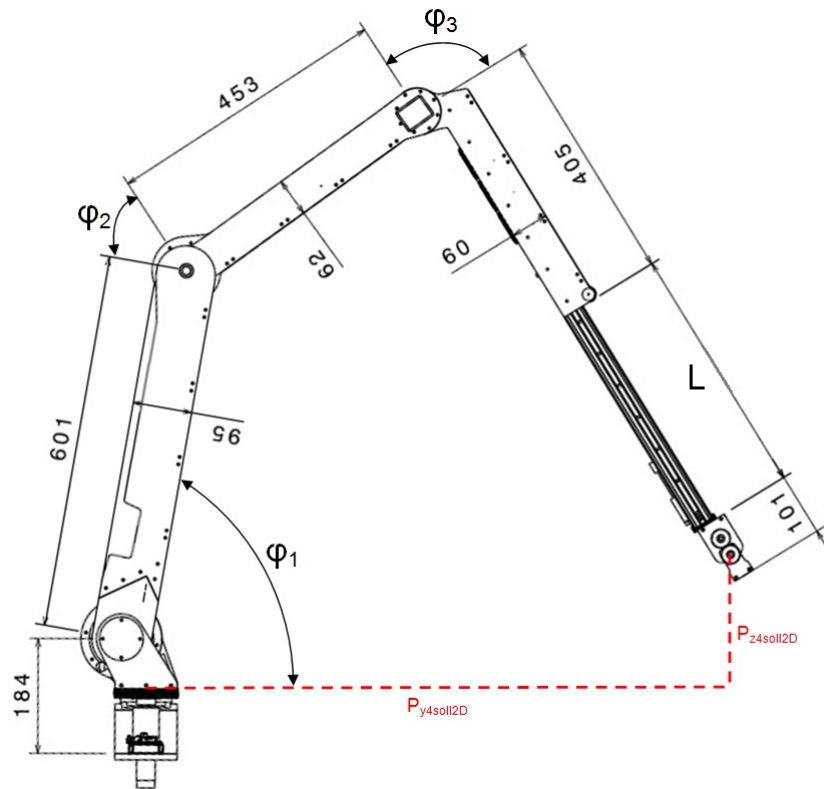


Abbildung 3.7: Reduziertes System (zweidimensional)

Für die Matrix  $\underline{T}_{World,4/2D}$  gilt nun

$$\underline{T}_{World,4/2D} = \underline{T}_{World,0/2D} + \underline{T}_{0,1/2D} + \underline{T}_{1,2/2D} + \underline{T}_{2,3/2D} + \underline{T}_{3,4/2D} =$$

$$\begin{bmatrix} -40 + 600,6 \cdot \cos(\varphi_1) + 450,1 \cdot \cos(\varphi_1 + \varphi_2) + (470,3 + L) \cdot \cos(\varphi_1 + \varphi_2 + \varphi_3) + 59,9 \cdot \sin(\varphi_1 + \varphi_2 + \varphi_3) \\ 80 + 59,9 \cdot \cos(\varphi_1 + \varphi_2 + \varphi_3) + 600,6 \cdot \sin(\varphi_1) + 450,1 \cdot \sin(\varphi_1 + \varphi_2) + (470,3 + L) \cdot \sin(\varphi_1 + \varphi_2 + \varphi_3) \end{bmatrix}$$

Durch Gleichsetzen der Vektorkoordinaten der Matrix  $\underline{T}_{World,4/2D}$  mit den zuerst errechneten Werten  $P_{y4soll2D}$  und  $P_{z4soll2D}$  erhält man nun zwei Gleichungen der Form

$$P_{y4soll2D} = -40 + 600,6 \cdot \cos(\varphi_1) + 450,1 \cdot \cos(\varphi_1 + \varphi_2) + (470,3 + L) \cdot \cos(\varphi_1 + \varphi_2 + \varphi_3) + 59,9 \cdot \sin(\varphi_1 + \varphi_2 + \varphi_3)$$

$$P_{z4soll2D} = 80 + 59,9 \cdot \cos(\varphi_1 + \varphi_2 + \varphi_3) + 600,6 \cdot \sin(\varphi_1) + 450,1 \cdot \sin(\varphi_1 + \varphi_2) + (470,3 + L) \cdot \sin(\varphi_1 + \varphi_2 + \varphi_3).$$

Es ist leicht zu erkennen, dass dieses Gleichungssystem unbestimmt ist, da die zwei Gleichungen vier Unbekannte beinhalten. Aus diesem Grund wird der Winkel  $\varphi_1$  und die Linearantriebsstellung  $L$  über einen Algorithmus anhand der Sollposition des TCP 's vorbestimmt. Das System ist nun lösbar und wird wegen Nichtlinearität mit dem mehrdimensionalen Newtonverfahren gelöst.

Das mehrdimensionale Newtonverfahren ist definiert durch

$$f(x_0, y_0) + f_x(x_0, y_0) \cdot (x - x_0) + f_y(x_0, y_0) \cdot (y - y_0) = 0$$

$$g(x_0, y_0) + g_x(x_0, y_0) \cdot (x - x_0) + g_y(x_0, y_0) \cdot (y - y_0) = 0$$

wobei an dieser Stelle für nähere Informationen auf die Literatur verwiesen wird. [FPDDKS08]

Nun können durch Gleichsetzen der Transformationsmatrix  $\underline{T}_{World,TCPsoll}$  mit der Transformationsmatrix  $\underline{T}_{World,TCPist}$  noch zwei weitere Gleichungen aufgestellt werden

$$P_{Xsoll} = P_{Xist}$$

$$P_{Ysoll} = P_{Yist}$$

und die fehlenden Winkel  $\varphi_4$  und  $\varphi_5$  berechnet werden.  $\underline{T}_{World,TCPist}$  ist an dieser Stelle die in Kapitel 3.2 hergeleitete Transformationsmatrix (siehe Abbildung 3.3), berechnet mit den hier errechneten Winkel  $\varphi_0$  bis  $\varphi_5$  und der Linearantriebsstellung  $L$ .

### 3.3.5 Resultierender Fehler

Zur Ermittlung des resultierenden Fehlers (Abweichung der Orientierung bzw. Position vom Sollwert) kann nun auch noch die Transformationsmatrix  $\underline{T}_{World,TCPist}$  mit der Transformationsmatrix  $\underline{T}_{World,TCPsoll}$  verglichen werden. [PBS08]

$$\underline{T}_{World,TCPsoll} = \underline{T}_{World,TCPist}$$

$$\Rightarrow \begin{bmatrix} u_{xsoll} & v_{xsoll} & w_{xsoll} & P_{Xsoll} \\ u_{ysoll} & v_{ysoll} & w_{ysoll} & P_{Ysoll} \\ u_{zsoll} & v_{zsoll} & w_{zsoll} & P_{Zsoll} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} u_{xist} & v_{xist} & w_{xist} & P_{Xist} \\ u_{yist} & v_{yist} & w_{yist} & P_{Yist} \\ u_{zist} & v_{zist} & w_{zist} & P_{Zist} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Der Fehler  $\Delta x$  in x-Richtung kann dann zum Beispiel wie folgt berechnet werden

$$\Delta x = |P_{Xsoll} - P_{Xist}|$$

## Kapitel 4

# Fazit und Dank

Im Rahmen dieser Arbeit wurden die mathematische Modellierung des Systems und die Berechnung der Vorwärtskinematik des Roboterarms für den Rescue Roboter beschrieben. Auf diesen Informationen aufbauend wurde ein Lösungsansatz erstellt, um später die inverse Kinematik zu berechnen. Die hier berechnete inverse Kinematik ist lediglich eine Möglichkeit von vielen um auf die Achswinkel zurückzurechnen und beschreibt nur die mathematische Berechnung der Positionsfindung. An dieser Stelle möchte ich mich bei meinen Eltern bedanken, da sie nicht nur mein Studium ermöglicht haben, sondern auch ständig ein sehr großes Interesse an meiner Arbeit zeigten und mich so gut es ging unterstützten. Besonderer Dank gilt auch Herrn Prof. (FH) Dipl.-Ing. Dr. Burkhard Stadlmann, der meine Bachelorarbeit betreut hat und Herrn DI (FH) Raimund Edlinger, sowie auch Herrn Ing. Michael Zauner Bsc für die hervorragende Unterstützung über die gesamte Projektphase hinweg. Bei dieser Arbeit habe ich auf jeden Fall viel Neues gelernt, was mir in meinem weiteren Studienweg und auch im späteren Berufsleben sehr nützlich sein wird.

# Abbildungsverzeichnis

1.1	Rescue Roboter MARK; oben: alt bzw. unten: neu[FW11]	2
1.2	Rescue Arena [A.S08]	2
2.1	Denavit-Hartenberg Parametrisierung [Sch98]	6
3.1	Screenshot Mathematica Notebookfile	8
3.2	Anordnung und Orientierung der Frames im System	11
3.3	Berechnung der Vorwärtskinematik mit Mathematica	11
3.4	Schematische Darstellung des Roboterarms	12
3.5	Einteilung des Systems in Sektionen	13
3.6	Reduziertes System	14
3.7	Reduziertes System (zweidimensional)	15

# Literaturverzeichnis

- [Ang07] ANGELES, Jorge: *Fundamentals of Robotic Mechanical Systems*. Springer Verlag, 2007
- [A.S08] A.SMITH: *RoboCup Rescue Arena*. [http://en.wikipedia.org/wiki/File:RoboCup\\_Rescue\\_2008\\_German\\_open\\_test\\_arena.JPG](http://en.wikipedia.org/wiki/File:RoboCup_Rescue_2008_German_open_test_arena.JPG). Version: April 2008
- [FPDDKS08] FH-PROF. DI DR. KLAUS SCHIEFERMAYR, o.Univ.-Prof. Dr. Peter W.: *Skriptum Angewandte Mathematik Teil2*. 2008
- [FW11] FH-WELS, Robo Racing T.: *RRT Wels*. <http://rrt.fh-wels.at/>. Version: April 2011
- [PBS08] PROF. BRUNO SICILIANO, Prof. Oussama K.: *Handbook of Robotics*. Springer Verlag, 2008
- [Sch98] SCHMIDT, Wolfgang: *Verallgemeinerte inverse Kinematik [Diplom Arbeit]*. 1998
- [Sta07] STADLMANN, Prof. (FH) Dipl.-Ing. Dr B.: *Skriptum Handhabungstechnnik und Robotik*. 2007
- [Wik11] WIKI, RoboCup: *Robot League*. [http://wiki.robotcup.org/wiki/Robot\\_League](http://wiki.robotcup.org/wiki/Robot_League). Version: April 2011