Fakultät für Angewandte Informatik
Lehrstuhl für Produktionsinformatik
Prof. Dr.-Ing. Johannes Schilp

Seminararbeit

Verbesserung der Pfad- und Trajektorienplanung am UR5

Vorgelegt von:	Nik Julin Nowoczyn
Matrikelnummer:	8529776
Studiengang:	Ingenieurinformatik
Prüfer:	Prof. Dr.-Ing. Johannes Schilp
Betreuer:	Ludwig Vogt
Ausgabedatum:	01. 10. 2022
Abgabedatum:	31. 03. 2023

Aufgabenstellung

In der Industrie der heutigen Zeit werden zur Herstellung verschiedenster Produkte Industrieroboter eingesetzt. Die Aufgabe der Forschung ist es daher, die eingesetzten Verfahren zu optimieren und die Verwendung neuer Produktionsmethoden mithilfe dieser Roboter zu optimieren. In der Universität wird der Roboter UR5 von Universal Robotics unter anderem eingesetzt, um automatisierte additive Fertigung zu ermöglichen. Dabei sollen die gefertigten Bauteile automatisiert aus dem Druckbereich entfernt und an anderer Stelle platziert werden.
In dieser Arbeit soll die Steuerung des UR5 näher untersucht werden, die einige Probleme aufweist. Es wurde unter anderem festgestellt, dass beim Fahren zwischen zwei Positionen oftmals eine Selbstkollision auftritt und in Folge ein Notstopp durchgeführt werden muss. Es gilt, eine mögliche Ursache für dieses Problem zu identifizieren und eine Lösung zu präsentieren, die es ermöglicht Selbstkollision zu erkennen und bei gleichem Pfad zu umgehen.

Kurzfassung

In dieser Arbeit wird die Steuerung des UR5 näher untersucht und eine Ursache der häufig auftretenden Selbstkollisionen ermittelt. Zudem werden alternative Algorithmen erläutert und die Vor- und Nachteile der verschiedenen Methoden verglichen. Dabei wird eine direkte und schnelle Berechnungsmethode zur Bestimmung der Gelenkwinkel vorgestellt und die auftretenden Selbstkollisionen durch dadurch begründet, dass die Software des UR5 vermutlich stattdessen auf einer numerischen Methode der inversen Kinematik basiert. Es wird festgestellt, dass für jede Endeffektor-Transformation in der Regel bis zu acht mögliche Roboterstellungen existieren und ein Algorithmus vorgestellt, der ohne komplizierte Pfadplanung alle Lösungswege für eine gerade Bahn berechnen und auf Kollision überprüfen kann.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

URDF Unified Robot Description Format

ROS das Robot Operating System

RRT Rapidly Exploring Random Tree

1 Direkte Kinematik

Die direkte Kinematik ist dafür verantwortlich, aus den verschiedenen Winkeln und Positionen der Gelenke die Rotation und Position des Endeffektors im Raum zu berechnen. Dazu wird zunächst in jedem Gelenk ein Koordinatenursprung gelegt, der eine Nullstellung jedes Gelenks beschreibt. Um alle Gelenke in einer kinematischen Kette abzubilden, kann mit einem Parameter in den Freiheitsgraden des entsprechenden Gelenks eine Rechenvorschrift aufgebaut werden, um den Roboter zu beschreiben und die Position des Endeffektors schnell bestimmen zu können.
Zur Beschreibung von Transformationen wird in dieser Arbeit auf die Homogene Transformationsmatrix zurückgegriffen. Diese besteht aus einer Rotationsmatrix $R_{a,b}$, die die Rotation von System a in System b beschreibt und einem Translationsvektor $P_{a,b}$, der die Translation zwischen den zwei Systemen entspricht 1.1 [1, S. 28]. Um eine Inverse Transformation von System b in System a zu erhalten, kann die Inverse der Transformationsmatrix $T_{a,b}$ oder die Regel aus Gleichung 1.2 verwendet werden. Bei der Multiplikation zweier geeigneter Matrizen können mehrere Transformationen in einer Matrix dargestellt werden. Ein Beispiel hierfür findet sich in Gleichung 1.3.

$$T_{a,b} = \begin{bmatrix} R_{a,b} & P_{a,b} \\ 0^T & 1 \end{bmatrix} \tag{1.1}$$

$$T_{b,a} = (T_{a,b})^{-1} = \begin{bmatrix} (R_{a,b})^T & -(R_{a,b})^T P_{a,b} \\ 0^T & 1 \end{bmatrix} = \begin{bmatrix} R_{b,a} & -R_{b,a} P_{a,b} \\ 0^T & 1 \end{bmatrix} \tag{1.2}$$

$$T_{a,b} \cdot T_{b,c} = T_{a,c} \tag{1.3}$$

1.1 DH-Konvention

?? Quelle
Um Rotation und Translation eines Gliedes der kinematischen Kette darzustellen kann die sog. DH-Konvention oder auch DH-Transformation verwendet werden. Diese Konvention beschreibt ein standardisiertes Vorgehen wie die Transformation des nächsten Gelenks im Koordinatensystem des momentanen Gelenks ausgedrückt werden soll [3, 1]. Es gelten die folgenden Regeln:

1. Achse z_n liegt entlang der Bewegungsachse des Gelenks n

2. Achse x_n liegt auf der kürzesten Verbindung zwischen Achsen z_{n-1} und z_n.

3. Die y_n Achse wird rechtshändig ergänzt.

Dabei sind die Ursprünge der Gelenkkordinatensysteme oftmals nicht im Gelenkursprung, was Komplexität für die Berechnung von Transformationen verringert. Aus der Beziehung der zwei Koordinatensysteme können die DH-Parameter abgeleitet werden (siehe auch Abbildung 1.1):

- θ_n Winkel zwischen x_{n-1} und x_n mit Rotationsachse z_{n-1}
- d_n: Kleinster Abstand zwischen x_{n-1} und x_n
- a_n: Abstand zwischen den Achsen z_{n-1} und z_n
- α_n Winkel zwischen z_{n-1} und z_n mit Rotationsachse x_n

Dies entspricht den folgenden Transformationsmatrizen (Gleichungen 1.4, 1.5, 1.6, 1.7):

$$T_{\theta_n} = \begin{bmatrix} \cos(\theta_n) & -\sin(\theta_n) & 0 & 0 \\ \sin(\theta_n) & \cos(\theta_n) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.4}$$

$$T_{d_n} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.5}$$

$$T_{a_n} = \begin{bmatrix} 1 & 0 & 0 & a_n \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.6}$$

$$T_{\alpha_n} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha_n) & -\sin(\alpha_n) & 0 \\ 0 & \sin(\alpha_n) & \cos(\alpha_n) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.7}$$

Um nun Koordinatensystem $n-1$ in Koordinatensystem n zu überführen, kann die Transformationsmatrix $T_{n-1,n}$ verwendet werden (Gleichung 1.8). Für den beweglichen

Freiheitsgrad der Gelenke wird dann jeweils eine freie Variable festgelegt (θ_n für translatorische und d_n für rotatorische Gelenke).

$$T_{n-1,n}(\theta_n, d_n) \coloneqq T_{\theta_n} \cdot T_{d_n} \cdot T_{a_n} \cdot T_{\alpha_n} = \begin{bmatrix} \cos(\theta_n) & -\sin(\theta_n)\cos(\alpha_n) & \sin(\theta_n)\sin(\alpha_n) & a_n\cos(\theta_n) \\ \sin(\theta_n) & \cos(\theta_n)\cos(\alpha_n) & -\cos(\theta_n)\sin(\alpha_n) & a_n\sin(\theta_n) \\ 0 & \sin(\alpha_n) & \cos(\alpha_n) & d_n \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.8}$$

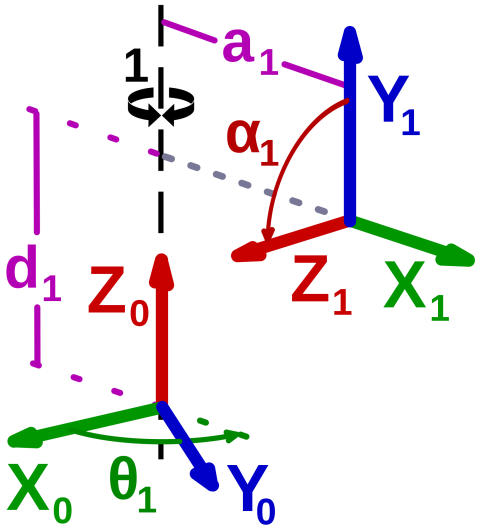

Abbildung 1.1: DH-Konvention zwischen zwei Gelenken [5]

Oftmals wird in der Literatur auch eine alternative, modifizierte Form der DH-Parameter verwendet [1, S. 75]. Um die Vergleichbarkeit mit [6] zu gewährleisten wird in der Rechnung sowie im Code diese alternative Methode eingesetzt. Dabei verändert sich die Transformationsmatrix eines Gelenks auf die Werte aus Gleichung 1.9:

$$T_{n-1,n}(\theta_n, d_n) \coloneqq \begin{bmatrix} \cos(\theta_n) & -\sin(\theta_n) & 0 & a_{n-1} \\ \sin(\theta_n)\cos(\alpha_{n-1}) & \cos(\theta_n)\cos(\alpha_{n-1}) & -\sin(\alpha_{n-1}) & -d_n\sin(\alpha_{n-1}) \\ \sin(\theta_n)\sin(\alpha_{n-1}) & \cos(\theta_n)\sin(\alpha_{n-1}) & \cos(\alpha_{n-1}) & d_n\cos(\alpha_{n-1}) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.9}$$

1.2 Unified Robot Description Format

Das Unified Robot Description Format (URDF) ist ein Standard, entwickelt für das Robot Operating System (ROS), der sowohl die geometrischen als auch physischen und visuellen Eigenschaften eines Roboters beschreiben kann. Dafür ist eine auf XML basierende Datei nötig, die mithilfe von XML-Tags den Roboter aus sog. „Links“ und „Joints“ aufbaut [7].
Links sind die physischen Verbindungen zwischen zwei Gelenken und können verschiedene Eigenschaften aufweisen. Neben den dem geometrischen Aufbau wird zudem unterschieden zwischen visuellen Eigenschaften („visual“), Kollisionseigenschaften („collision“) und Trägheitseigenschaften („inertial“).
Joints sind Gelenke, die aus der Verbindung zweier physischer Links bestehen. Dabei sind nicht nur translatorische („prismatic“) und rotatorische Gelenke („revolute“) beschreibbar, sondern auch feste („fixed“), schwebende („floating“) und planare Verbindungen („planar“). Für die Beschreibung eines Joints wird der vorhergehende Link als „parent“ und der nächste Link als „child“ bezeichnet. Zudem müssen im Feld „origin“ Translation und Rotation des Ursprungs im Koordinatensystem des Parent Links, sowie im Feld „axis“ je nach Gelenk die Rotationsachse, Translationsachse oder Normale der Bewegungsoberfläche angegeben werden. Desweiteren ist es möglich, Schnittstellen für die Bewegung der Motoren zu definieren und Bewegungslimits für die Gelenke anzugeben, um die Ansteuerung des Roboters und die Bewegungsplanung zu vereinfachen.
Die Schritte zur Berechnung der direkten Kinematik können für das bessere Verständnis im Code ?? (Anhang referenz) direkt nachvollzogen und visualisiert werden.

1.3 Beschreibung des UR5

Der in dieser Arbeit betrachtete Roboter ist der UR5e von Universal Robots. Dieser ist ein vergleichsweise günstiger Roboter mit verringerter Zahl an Singularitäten (siehe Abschnitt 2.4), sechs Gelenken und einer offenen kinematischen Kette. Offiziell wird der Roboter mit den DH-Parametern aus Tabelle 1.1 und dynamischen Eigenschaften der Links aus Tabelle 1.2 beschrieben [8]. Abbildung 1.2 kann zudem die Dimensionierung des Roboters und die Anordnung der Achsen in Nullstellung entnommen werden.
Nach Datenblatt [9] hat der UR5 einen Arbeitsraum mit Radius 85cm (Abstand vom Befestigungspunkt bei ausgestrecktem Arm) und kann sich mit einer Maximalgeschwindigkeit von bis zu 180°/s bewegen.

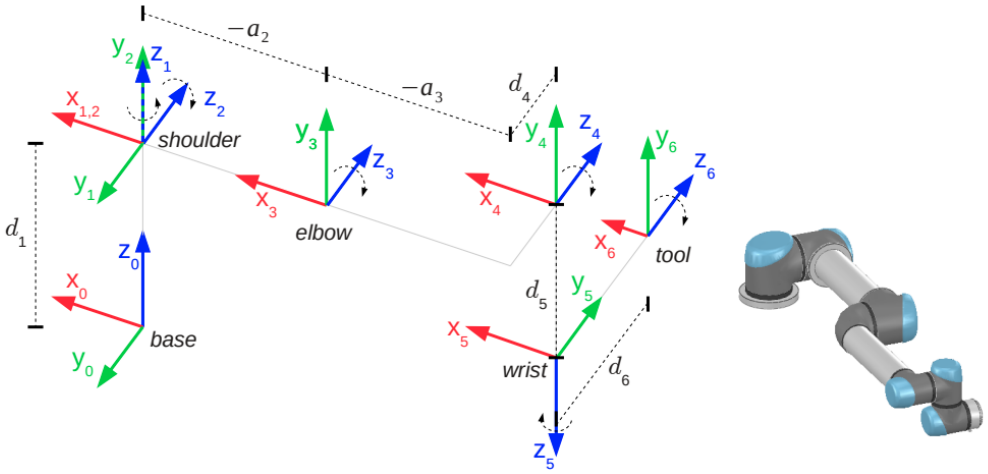

Abbildung 1.2: Achsen des UR5-Roboters mit Angabe der DH-Parameter (links) und Visualisierung der Nullposition (rechts) [6]

Tabelle 1.1: DH-Parameter des UR5e Roboters von Universal Robots [8]

Joint	$\boldsymbol{\theta}$ **[rad]**	$\boldsymbol{a}$ **[m]**	$\boldsymbol{d}$ **[m]**	$\boldsymbol{\alpha}$ **[rad]**
Joint 1	0	0	0.089159	$\pi/2$
Joint 2	0	-0.425	0	0
Joint 3	0	-0.3922	0	0
Joint 4	0	0	0.10915	$\pi/2$
Joint 5	0	0	0.09465	$-\pi/2$
Joint 6	0	0	0.0823	0

1.4 Geschwindigkeitskinematik

Um einen Roboter in Bewegung darzustellen, ist nicht nur die Transformation der Gelenke relevant, sondern auch dessen Kinematik und Dynamik. Dies ist besonders relevant, da in der echten Welt bestimmte Maximalwerte für (Winkel-) Geschwindigkeit und Beschleunigung, sowie Trägheit und Drehmoment eingehalten werden müssen, um den sicheren Betrieb sowie das Verhalten der Masse korrekt abzubilden.

Für die Berechnung der Geschwindigkeit des Endeffektors im Gelenk e kann die Jakobi-Matrix verwendet werden. Dabei gilt für eine Drehung nach DH-Konvention um Achse z_{n-1} für die Geschwindigkeit Gleichung 1.10 und für die Winkelgeschwindigkeit Gleichung 1.11. Für translatorische Gelenke gelten stattdessen Gleichungen 1.12 und 1.13. Die Vektoren z_i entsprechen jeweils der Einheitsvektor der z-Achse $z_i = \frac{Z_{0,i}}{|Z_{0,i}|}$ auf Basis der Transformationsmatrix $T_{0,i} = \prod_{n=1}^{i} T_{n-1,n}$ (siehe Konvention Gleichung 2.4).

Tabelle 1.2: Beschreibung der dynamischen Eigenschaften des UR5e Roboters von Universal Robots [8]

Link	**Masse [kg]**	**Schwerpunkt [m]**
Link 1	3.7	[0, -0.02561, 0.00193]
Link 2	8.393	[0.2125, 0, 0.11336]
Link 3	2.33	[0.15, 0.0, 0.0265]
Link 4	1.219	[0, -0.0018, 0.01634]
Link 5	1.219	[0, 0.0018,0.01634]
Link 6	0.1879	[0, 0, -0.001159]

$$J_{P_n} = z_{n-1} \times (P_e - P_{n-1}) \tag{1.10}$$

$$J_{\omega_{n+1}} = z_{n-1} \tag{1.11}$$

$$J_{P_n} = z_{n-1} \tag{1.12}$$

$$J_{\omega_{n+1}} = 0 \tag{1.13}$$

Für alle Gelenke ergibt sich dann für die Jakobi-Matrix eines Sechs-Achsen-Roboters wie dem UR5 die folgende Gleichung 1.14 mit jeweils sechs Spalten und Zeilen. Um die Geschwindigkeit des Endeffektors v_6 zu erhalten, muss die Matrix mit der Geschwindigkeit $\dot{\theta}$ multipliziert werden (Gleichung 1.15).

$$J_{0,6} = \begin{pmatrix} J_{P_1} & \dots & J_{P_6} \\ J_{\omega_1} & \dots & J_{\omega_6} \end{pmatrix} \tag{1.14}$$

$$v_6 = \begin{pmatrix} \dot{P}_{0,6} \\ \omega_6 \end{pmatrix} = J_{0,6} \cdot \dot{\theta} \tag{1.15}$$

2 Inverse Kinematik

In der direkte Kinematik wird aus den Gelenkzuständen Position und Rotation des Endeffektors bestimmt. Der umgekehrte Vorgang, also das Berechnen der Gelenkpositionen bei gegebener Zieltransformation, wird als inverse Kinematik bezeichnet. Dies ist erforderlich, um Aufgaben in der Umgebung des Roboters zu lösen und die angestrebten Zielpunkte zu erreichen. Dabei muss beachtet werden, dass oft mehrere gleichwertige Lösungen für eine Zielstellung des Endeffektors vorliegen und die inverse Kinematik offener Ketten oft nicht einfach direkt mithilfe einer mathematischen Formel gelöst werden kann (siehe Abschnitt 2.1). Um die Gelenkwinkel zu erhalten, kann eine Lösung prinzipiell analytisch, numerisch oder geometrisch ermittelt werden. Die Vor- und Nachteile, sowie die Berechnung dieser Methoden wird in diesem Kapitel näher erläutert.

2.1 Analytische Lösung

Für eine direkte Lösung muss zunächst die Formel aus Gleichung 1.8 oder 1.9 herangezogen werden. Nach Multiplikation der Transformationsmatrizen müsste die entstehende Gleichung 2.1 nach den freien Variablen θ_1 bis θ_n bzw. d_1 bis d_n aufgelöst werden. Da aufgrund der rotatorischen Gelenke nichtlineare trigonometrische Funktionen verwendet werden, ist diese Funktion bei Industrierobotern überwiegend nichtlinear und in vielen Fällen auch nicht lösbar.

$$T_{0,n}(\overrightarrow{q}) = \prod_{i=1}^{n} T_{i-1,i}(q_i) \tag{2.1}$$

Im Falle des UR5 ist dieser Ansatz aufgrund der oben genannten Einwände nicht anwendbar und wird an dieser Stelle deshalb nicht detailliert ausgeführt.

2.2 Numerische Lösung

?? Quelle

Mit numerischer Approximation können Gelenkwinkeländerungen mit vergleichsweise geringem Rechenaufwand bestimmt werden. Zur Berechnung der Gelenkwinkel des nächsten Zeitschritts wird die Geschwindigkeitskinematik aus Abschnitt 1.4 herange-

zogen und mithilfe der Jakobi-Matrix eine Winkeländerung relativ zur momentanen Transformation approximiert. Nach Umformen von Gleichung 1.15 in Gleichung 2.2 kann θ durch lineare Approximation näherungsweise bestimmt werden (siehe Gleichung 2.3). Dabei sind dx, dy und dz die Distanz zwischen Startpunkt (Zeitpunkt t) und Zielpunkt (Zeitpunkt $t+1$), sowie $d\alpha$, $d\beta$ und $d\gamma$ die Differenzen der Eulerwinkel der Matrix $R_{0,6}$.

$$\dot{\theta} = J^{-1} \cdot v_6 = \begin{pmatrix} \dot{P}_{0,6} \\ \omega_6 \end{pmatrix}^{-1} \cdot v_6 \tag{2.2}$$

$$\theta_{t+1} = \theta_t + \dot{\theta} \begin{bmatrix} dx \\ dy \\ dz \\ d\alpha \\ d\beta \\ d\gamma \end{bmatrix} \tag{2.3}$$

Da die Ergebnisse auf den vorherigen Winkelstellungen aufbauen und um ein sinnvolles Ergebnis zu erhalten, müssen die Differenzen entsprechend möglichst klein sein. In [2] wurde hierzu experimentell ermittelt, dass beim UR5 ein Abstand von $40\mu m$ über eine Trajektorie der Länge $1m$ in jede Richtung einen Fehler von bis zu $0.1mm$ verursachen kann. Zudem fällt auf, dass in diesem Verfahren wird nur eine einzige Lösung für eine Trajektorie generiert wird. Dies kann deshalb unter Umständen dazu führen, dass ein ungeeigneter Pfad durch den Konfigurationsraum gewählt und eine Selbstkollision herbeigeführt wird (siehe Abschnitt 3.1). Es ist möglich, dass diese Berechnungsmethodik auch in der installierten Steuerungssoftware des UR5 verwendet wird und der UR5 deshalb beim Anfahren von Zielkoordinaten oft in Situationen gerät, bei denen eine Selbstkollision nicht mehr vermeidbar ist und ein Notstopp eingeleitet wird.

2.3 Geometrische Lösung

Die schnellste Berechnung der Gelenkwinkel wird in einer direkten analytischen Lösung erreicht. Da dies aber oft nicht möglich ist, kann mithilfe der geometrischen Eigenschaften des jeweiligen Roboters eine geometrische Lösung ausgearbeitet werden. Diese Berechnung ist dann immer noch direkt, allerdings kann sie nicht universell für alle Roboter gleichsam eingesetzt werden. Im Falle des UR5 ist die Berechnung aufgrund des ähnlichen Aufbaus nur auf den UR5e und den UR10 nach Änderung der DH-Parameter übertragbar.

Zudem wird oft, um die Berechnung zu vereinfachen, in den äußeren drei Gelenken eines Sechsarmroboters ein Handwurzelpunkt eingeführt, in dem sich die Achsgeraden dieser Gelenke schneiden. Da die Drehung des sogenannten Handgelenks die Position des Handwurzelpunktes nicht verändert, kann so zuerst die Berechnung der Position des Handwurzelpunkts mithilfe der Zieltransformation durchgeführt werden und im Anschluss unabhängig die Berechnung der restlichen Gelenke stattfinden. Beim UR5 liegt kein Handwurzelpunkt vor, was den Vorteil hat, dass weniger Singularitäten vorliegen als bei vergleichbaren Robotern. Obwohl die Berechnung der Inversen Kinematik erschwert ist, kann ein ruhigeres und mehr vorhersehbares Bewegungsverhalten erwartet werden (siehe Abschnitt 2.4).

Dennoch kann durch eine Analyse der geometrischen Eigenschaften und der geringen Zahl der Singularitäten des UR5 eine Lösung von $\overrightarrow{\theta}$ gefunden werden. Die Strategie hierbei ist, beim ersten Gelenk zu beginnen und nach und nach die anderen Gelenke mit der Transformation $T_{0,6}$ des Endeffektors zu verknüpfen. Bekannt sind zu Beginn der Rechnung nur die DH-Parameter $(\theta_i, a_i, d_i, \alpha_i)$ jedes Gelenks i im Ausgangszustand, wobei θ_i als freie Variable jedes Gelenks betrachtet wird (Abschnitt 1.3), sowie die Transformation des Endeffektors $T_{0,6}$. Eine detailliertere Rechnung kann [6] und [4] entnommen werden.

In der folgenden Rechnung gilt stets die untenstehende Konvention aus [1, S. 82] für eine beliebige Transformation $T_{a,b}$ von einem System a in ein System b (Gleichung 2.4), sowie die in Abschnitt 1.1 vorgestellten Rechenregeln für Transformationsmatrizen.

$$T_{a,b} = \begin{bmatrix} X_{a,b} & Y_{a,b} & Z_{a,b} & P_{a,b} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} X_{a,bx} & Y_{a,bx} & Z_{a,bx} & P_{a,bx} \\ X_{a,by} & Y_{a,by} & Z_{a,by} & P_{a,by} \\ X_{a,bz} & Y_{a,bz} & Z_{a,bz} & P_{a,bz} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

1. Gelenk eins (Basis)

Zunächst wird der Ausgangspunkt $P_{0,5}$ von Gelenk 5 berechnet (Gleichung 2.5 [6, S. 4]) und im Anschluss trigonometrisch Winkel θ_1 bestimmt (Gleichung 2.6 sowie Abbildung 2.1). Dabei enstehen zwei Lösungen, die die Schulter des Roboters entweder links oder Rechts vom Ursprung platzieren.

$$P_{0,5} = T_{0,6} \cdot \begin{bmatrix} 0 \\ 0 \\ -d_6 \\ 1 \end{bmatrix} \tag{2.5}$$

$$\theta_1 = \arctan2(P_{0,5y}, P_{0,5x}) \pm \arccos\left(\frac{d_4}{\sqrt{P_{0,5x}^2 + P_{0,5y}^2}}\right) + \frac{\pi}{2} \tag{2.6}$$

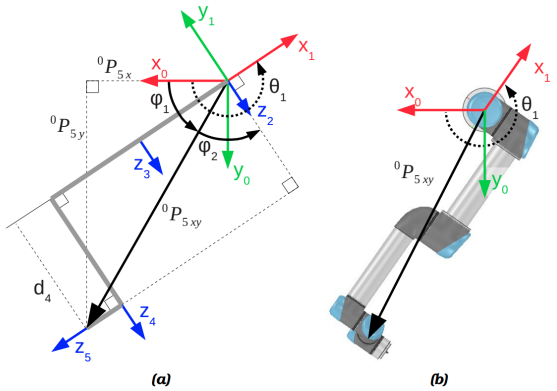

Abbildung 2.1: Berechnung von θ_1, Betrachtung von Gelenk eins bis fünf [6]

2. Gelenk fünf (oberes Handgelenk)

Im Anschluss kann θ_5 bestimmt werden, da der y-Teil der Position von Gelenk 6 relativ zu Gelenk 1 ($P_{1,6y}$) nur mithilfe von θ_5 und bereits bekannter Parameter $P_{0,6}$, θ_1 und der DH-Parameter berechnet werden kann (siehe Abbildung 2.2). Aus dieser Erkenntnis ergibt sich Gleichung 2.7 $P_{1,6y}$ erhält man durch die Rotation des Ursprungssystems $T_{0,6}$ um die z_1-Achse (Gleichung 2.8). In Kombination erhält man die Gleichung für θ_5 (Gleichung 2.9). Dabei entstehen wiederum zwei Lösungen, die jeweils das Handgelenk ober- oder unterhalb des Arms platzieren.

$$-P_{1,6y} = d_4 + d_6 \cdot \cos(\theta_5) \tag{2.7}$$

$$P_{1,6y} = -P_{0,6x} \cdot \sin(\theta_1) + P_{0,6y} \cdot cos(\theta_1) \tag{2.8}$$

$$\theta_5 = \pm \arccos\left(\frac{P_{0,6x} \cdot \sin\theta_1 - P_{0,6y} \cdot \cos\theta_1 - d_4}{d_6}\right) \tag{2.9}$$

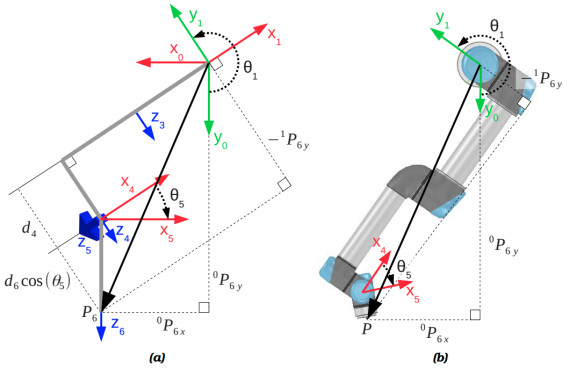

Abbildung 2.2: Berechnung von θ_5, Betrachtung von allen Gelenken [6]

3. Gelenk sechs (Endeffektor)

Nach θ_1 und θ_5 wird θ_6 bestimmt. Dazu wird die Eigenschaft des UR5 herangezogen, dass die Z-Achsen von Gelenken zwei, drei und vier stets parallel zur Y-Achse von Gelenk 1 verlaufen (siehe Abbildung 1.2). Deshalb kann die Y-Achse y_1 beschrieben von Gelenk sechs ($Y_{6,1}$) aus unabhängig von $\theta_{1,2,3,4}$ betrachtet und mithilfe von sphärischen Koordinaten als Funktion $f(r, \gamma, \phi) = -Y_{6,1}(-\theta_6, \theta_5)$ aufgefasst werden (Radius $r = 1$, $-\theta_6$ als Azimuth γ, sowie θ_5 als polaren Winkel ϕ; siehe Abbildung 2.3). Eine Umrechnung in kartesische Koordinaten ergibt deshalb Gleichung 2.10, wobei $Y_{6,1}$ mithilfe einer Drehung mit Winkel θ_1 um die Achse z_1 beschrieben werden kann (Gleichung 2.11). Z und X entsprechen hierbei den Werten der Konvention 2.4. Nach Gleichsetzen der x- und y-Einträge der beiden Gleichungen und anschließendem Umformen erhält man θ_6 (Gleichung 2.12). Dabei kann es genau eine Lösung geben. Falls $\sin\theta_5 = 0$, liegt eine Singularität vor und eine Lösung kann nicht bestimmt werden. Dies tritt auf, wenn neben den Achsen z_2, z_3 und z_4 auch Achse z_6 parallel steht. In diesem Fall kann eine beliebige Lösung gewählt werden, beispielsweise $\theta_6 = 0$.

$$Y_{6,1} = \begin{bmatrix} -\sin\theta_5 \cdot \cos\theta_6 \\ \sin\theta_5 \cdot \sin\theta_6 \\ -\cos\theta_5 \end{bmatrix} \tag{2.10}$$

$$Y_{6,1} = -\sin\theta_1 \cdot X_{6,0} + \cos\theta_1 \cdot Y_{6,0} \tag{2.11}$$

$$\theta_6 = \arctan2\left(\frac{-X_{6,0y} \cdot \sin\theta_1 + Y_{6,0y} \cdot \cos\theta_1}{\sin\theta_5}, \frac{X_{6,0x} \cdot \sin\theta_1 - Y_{6,0x} \cdot \cos\theta_1}{\sin\theta_5}\right) \tag{2.12}$$

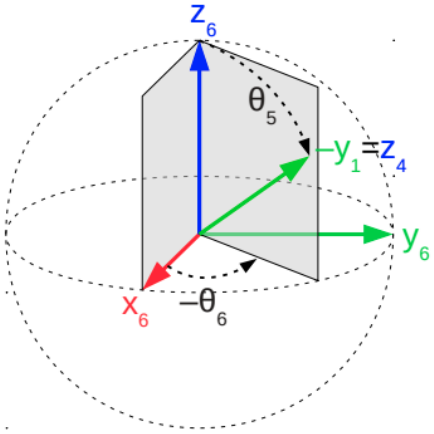

Abbildung 2.3: Berechnung von θ_6 mit Sphärischen Koordinaten, für Koordinatensystembeschreibung siehe Abbildung 1.2 [6]

4. Gelenk drei (Ellenbogen)

Die verbleibenden drei Gelenke haben alle parallele Achsen und lassen sich dadurch zu einem zweidimensionalen System vereinfachen (siehe Abbildung 2.4, rechts). Um θ_3 zu berechnen kann der Winkel ϕ_3 zu Hilfe genommen werden (siehe Gleichung 2.13). Die Werte a_2, a_3 sind dabei die DH-Parameter der jeweiligen Gelenke und $|P_{1,4xz}|$ ist der Abstand zwischen Gelenk 1 und Gelenk 4, dessen relative Position bereits bekannt ist. Aufgrund der Geometrie gilt: $|P_{1,4xz}| \in |a_2 \pm a_3|$ Nach der Anwendung des arccos erhält man in der Regel zwei Lösungen, die der Position „Elbow Up“ und „Elbow Down“ entsprechen (Gleichung 2.14).

$$\cos(\theta_3) = -cos(\phi_3) = \frac{a_2^2 + a_3^2 - |P_{1,4xz}^2|}{2a_2a_3} \tag{2.13}$$

$$\theta_3 = \pm \arccos\left(\frac{a_2^2 + a_3^2 - |P_{1,4xz}^2|}{2a_2a_3}\right) \tag{2.14}$$

5. Gelenk zwei (Schulter)

θ_2 kann mithilfe von Abbildung 2.4 ($\theta_2 = \phi_1 - \phi_2$), sowie Gleichung 2.15 und 2.16 bestimmt werden. Nach der Substitution von $\phi_3 = \sin(\pi - \theta_3) = \sin(\theta_3)$ erhält man eine Lösung für θ_2 (Gleichung 2.17).

$$\phi_1 = \arctan2(-P_{1,4_z}, -P_{1,4x}) \tag{2.15}$$

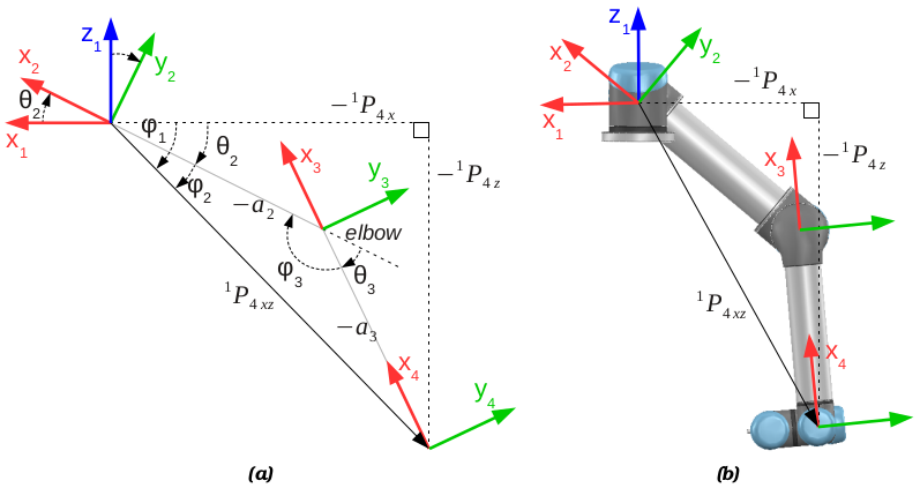

Abbildung 2.4: Berechnung von θ_4 durch Vereinfachung als zweidimensionales System und Bestimmung der Transformation zwischen Gelenk 1 und Gelenk 4 [6]

$$\phi_2 = \arcsin\left(\frac{-a_3 \cdot \sin\phi_3}{|P_{1,4xz}|}\right) \tag{2.16}$$

$$\theta_2 = \phi_1 - \phi_2 = \operatorname{arctan2}(-P_{1,4_z}, -P_{1,4x}) - \arcsin\left(\frac{-a_3 \cdot \sin\theta_3}{|P_{1,4xz}|}\right) \tag{2.17}$$

6. Gelenk vier (unteres Handgelenk)

Der Winkel des letzten Gelenks ist definiert als der Winkel zwischen den Achsen x_3 und x_4 um Rotationsachse z_4 (siehe Definition DH-Parameter, Abschnitt 1.1). Da alle anderen Winkel bereits bekannt sind, kann der letzte Winkel der ersten Spalte $X_{3,4}$ der Transformationsmatrix $T_{3,4}$ entnommen werden. Mit den ersten beiden Werten dieser Spalte, $X_{3,4x}$ und $X_{3,4y}$, erhält man den Wert von θ_4 (Gleichung 2.18)

$$\theta_4 = \operatorname{arctan2}\left(X_{3,4y}, X_{3,4x}\right) \tag{2.18}$$

Zusammenfassung

Mithilfe der Gleichungen 2.19 bis 2.24 kann die Inverse Kinematik eines UR5-Roboters direkt bestimmt werden. Dabei existieren in der Regel acht mögliche Lösungen, jeweils eine für die Gelenke sechs (Endeffektor), zwei (Schulter) und vier (unteres Handgelenk) und jeweils zwei für die Gelenke eins (Basis), drei (Ellenbogen) und fünf (oberes Handgelenk). Falls die Achsen z_2, z_3, z_4 und z_6 parallel stehen, kann θ_6 beliebig gewählt werden, sodass eine Singularität mit beliebig vielen Lösungen vorliegt. Neben

den DH-Parametern und der gewünschten Transformation des Endeffektors muss für Gleichung 2.19 $P_{0,5}$, für Gleichung 2.21 die Transformation $T_{6,0}$, für Gleichung 2.22 und 2.22 die Transformation $T_{1,4}$ und für Gleichung 2.24 die Transformation $T_{3,4}$ mit der Regeln aus Gleichungen 1.1, 1.2 und 1.3 bestimmt werden.

$$\theta_1 = \arctan2(P_{0,5y}, P_{0,5x}) \pm \arccos\left(\frac{d_4}{\sqrt{P_{0,5x}^2 + P_{0,5y}^2}}\right) + \frac{\pi}{2} \tag{2.19}$$

$$\theta_5 = \pm \arccos\left(\frac{P_{0,6x} \cdot \sin\theta_1 - P_{0,6y} \cdot \cos\theta_1 - d_4}{d_6}\right) \tag{2.20}$$

$$\theta_6 = \arctan2\left(\frac{-X_{6,0y} \cdot \sin\theta_1 + Y_{6,0y} \cdot \cos\theta_1}{\sin\theta_5}, \frac{X_{6,0x} \cdot \sin\theta_1 - Y_{6,0x} \cdot \cos\theta_1}{\sin\theta_5}\right) \tag{2.21}$$

$$\theta_3 = \pm \arccos\left(\frac{a_2^2 + a_3^2 - |P_{1,4xz}^2|}{2a_2a_3}\right) \tag{2.22}$$

$$\theta_2 = \phi_1 - \phi_2 = \arctan2(-P_{1,4_z}, -P_{1,4x}) - \arcsin\left(\frac{-a_3 \cdot \sin\theta_3}{|P_{1,4xz}|}\right) \tag{2.23}$$

$$\theta_4 = \arctan2\left(X_{3,4y}, X_{3,4x}\right) \tag{2.24}$$

Die Schritte zur Berechnung der inversen Kinematik können für das bessere Verständnis im Code direkt nachvollzogen und visualisiert werden. (?? Referenz)

2.4 Singularitäten

Wie in Abschnitt 2.3 ermittelt, existiert beim UR5 genau eine Position, bei der eine Singularität vorliegt. Singularitäten zeichnen sich dadurch aus, dass es unendlich viele Lösungen für eine bestimmte Zieltransformation gibt. In der Regel tritt dies genau dann auf, wenn zwei Rotationsachsen parallel zueinander stehen und ihre Bewegung durch das Drehen in gegensätzliche Richtungen beliebig ausgleichen können.
In der Praxis tritt dieser Fall allerdings kaum auf, da durch kleine Rundungsfehler bei der Rechnung nur im Extremfall eine exakte Parallelität hergestellt wird. Trotzdem sind Singularitäten problematisch, da beim Durchfahren einer solchen Position extreme Positionsänderungen der verschiedenen Gelenke erforderlich sind. Aus diesem Grund ist es ratsam, diese Positionen möglichst zu vermeiden. Für weitere Informationen zur Pfadplanung, siehe Abschnitt 3.

3 Pfadplanung

Pfadplanungsalgorithmen werden häufig in der Robotik und der Automatisierungstechnik eingesetzt, um Pfade für Roboter oder Maschinen zu beschreiben, die sich innerhalb eines bestimmten Arbeitsbereichs bewegen. Dabei werden eine Reihe an Konfigurationen zu berechnet, die den Weg von einer bestimmten Transformation in eine andere beschreiben. Eine einzelne Konfiguration ist im Falle des UR5 ein Tupel mit sechs Winkeln, das die Ausrichtung der sechs Gelenke des Roboters angibt.
Die einfachste Form der Pfadplanung ist der kürzeste Weg zwischen zwei Punkten (Abschnitt 3.2), was aber nicht der schnellsten Verbindung entspricht. Die schnellste Verbindung ist immer der Weg, der beschritten wird, wenn bei konstanter maximaler Winkelgeschwindigkeit zwischen beiden Transformationen gewechselt wird (Abschnitt 1). Komplizierter wird es, wenn Beschränkungen im Arbeitsraum des Roboters, wie Kollisionen mit der Umgebung oder mit sich selbst, vorliegen.
In diesem Kapitel sollen die Grundlagen der Pfadplanung, sowie einige Algorithmen näher vorgestellt werden. Zudem wird in Abschnitt 3.3 eine einfache Implementierung der Pfadplanung vorgestellt, die ein Abfahren von Endeffektor-Stellungen ohne Selbstkollision ermöglicht.

3.1 Konfigurationsraum

Der Konfigurationsraum C stellt die Grundlage für die Algorithmen in der Pfadplanung dar und beschreibt alle möglichen Konfigurationen, die der Roboter einnehmen kann. Eine Konfiguration des UR5 ist dabei wie in Gleichung 3.1 definiert, wobei die Winkelwerte θ_i durch ihre Maximal- und Minimalwerte des Datenblatts [9] von $\pm 360°$ beschränkt sind. Gleichsam existieren Konfigurationen innerhalb dieser Menge $O \in C$, die Kollisionen beschreiben. Darunter zählen Konfigurationen, bei denen sich der Roboter selbst im Weg steht, sowie welche, bei denen der Untergrund oder statische Hindernisse eine erfolgreiche Ausführung der Bewegung verhindern. Der für die Pfadplanung relevante Bereich ist demzufolge der in Gleichung 3.2 beschriebene Menge C_{free}, der freie Konfigurationsraum, dessen Konfigurationen den Bereich beschreiben, durch den Roboter sich bewegen kann.

$$C = \{(\theta_1, \ldots, \theta_6) \in \theta_1 \times \cdots \times \theta_6 \mid \theta_i \in [-2\pi, 2\pi]\} \tag{3.1}$$

$$C_{free} = C \backslash O \tag{3.2}$$

Je nach Algorithmus muss der Konfigurationsraum allerdings nicht im vorhinein berechnet werden, insbesondere wenn der Konfigurationsraum sehr groß ist oder dynamisch Hindernisse erkannt werden sollen.
??Quelle

3.2 Berechnungsmethoden

Um den Roboter auf dem kürzesten Weg zu einer Zielposition zu bringen, wird die Strecke zunächst in mehrere Zwischenstufen zwischen zwei oder mehr Transformationspunkten T_1 und T_2 unterteilt. Jeder Schritt kann dabei mithilfe des Faktors $t \in [0, 1]$ beschrieben werden (Gleichung 3.3). Zwischen zwei Rotationspunkten R_1 und R_2 wird in der Regel der sogenannte Slerp verwendet (Gleichung 3.4), der jedoch erst in Quaternionen (Q1 und Q2) umgerechnet werden muss. Zu beachten sind zudem die Rechenregeln der Quaternionen, die hier nicht näher erklärt werden.

$$P_t = P_0 + (P_1 - P_0) \cdot t \tag{3.3}$$

$$Q_t = Q_1 \cdot \left(Q_1^{-1} \cdot Q_2\right)^t \tag{3.4}$$

Die resultierende Liste von Endeffektor-Transformationen muss dann in den Konfigurationsraum des Roboters übertragen werden. Der UR5 hat, wie in Abschnitt 2.3 berechnet, für jede Zwischentransformation oftmals bis zu acht, oder in einer Singularität sogar unendlich viele Stellungen für die Gelenke. Um einen geeigneten Pfad durch den Konfigurationsraum zu nehmen und die Selbstkollision sowie das Erreichen von Winkelbegrenzungen zu vermeiden, ist die Verwendung eines Pfadplanungsalgorithmus ratsam.
?? Quelle

3.2.1 Zellendekomposition

?? Quelle
In der Zellendekomposition wird der freie Konfigurationsraum C_{free} in kleinere Felder, sog. Zellen unterteilt. Zwei Konfigurationen liegen genau dann in der gleichen Zelle, falls ein Übergang zwischen den beiden Zuständen kollisionsfrei möglich ist und zwei benachbarte Zellen müssen über einen einfachen Pfad miteinander kollisionsfrei verbunden sein. Dies setzt konvexe Zellen voraus, bei denen jeder Punkt innerhalb der Zelle

jeden anderen erreichen kann. Auf Basis dieser Gruppierungen kann dann ein Konnektivitätsgraph oder eine Roadmap erstellt werden. Als Datenstruktur kann ein KD-Tree verwendet werden, der ein Konfigurationsraum als Baum so abbildet, dass Abstände und benachbarte Zellen zügig berechnet werden können. Im zweidimensionalen Raum werden hierfür die Positionen und Dimensionen von Quadraten sowie dessen Nachbarn gespeichert.
Diese direkte Berechnung der Zellenstruktur ist im sechsdimensionalen Fall des UR5 allerdings nichttrival und aufwändig. Aus diesem Grund und für die schnellere Berechnung wird deshalb oftmals eine approximierte Lösung aus einem Sampling-Verfahren bevorzugt.

3.2.2 Sampling-Verfahren

?? Quelle
In Sampling-basierten Verfahren wird das Gruppieren aller Konfigurationen in konkave Zellen übersprungen und stattdessen nur einzelne, benötigte Konfigurationen auf Kollision geprüft. Das bedeutet, dass nicht die beste Lösung aller möglichen Lösungen ermittelt wird, sondern nur die beste aller bereits ermittelten Lösungen. Der Algorithmus kann dementsprechend bereits nach dem Finden der ersten Lösung terminieren. Bedingung dafür ist allerdings eine Möglichkeit, eine Konfiguration schnell auf Kollisionsfreiheit zu überprüfen zu können. Insofern der Konfigurationsraum vollständig bekannt ist oder schon zuvor berechnet wurde, kann alternativ auch ein Lookup-Table verwendet werden.
Im Folgenden wird zwischen Single- und Multi-Query-Verfahren unterschieden:

Single-Query Verfahren

Bei Single-Query-Verfahren wird für jede Abfrage ein neuer Weg gesucht, bei Multi-Query-Verfahren werden vergangene Anfragen als Hilfestellung für neue Anfragen genutzt und so eine Roadmap erzeugt. Üblicherweise wird beim Single-Query-Verfahren mit Startkonfiguration q_a und Zielkonfiguration q_b wie folgt vorgegangen:

1. Initalisierung eines Graphens $G(V, E)$ mit Knoten mit $V = \{q_a, q_b\}$ und $E = \{\}$
2. Auswahl eines Expansionsknotens aus V mit einem gewählten Algorithmus
3. Berechnung eines neuen Knotens und einer Kante mit einem gewählten Algorithmus (inklusive Kollisionsüberprüfung)

4. Erweiterung des Graphens mit zuvor gefundenem Knoten und neuer Kante (falls möglich).

5. Prüfe auf Lösung in G (Verbindung von Start und Zielknoten) oder wiederhole ab 2.

Zudem kann der Algorithmus auf zwei oder mehr Suchbäume ausgeweitet werden, die sich nach einiger Zeit in der Mitte treffen. Dazu wird dann in den Schritten 24 der verwendete Baum zufällig ausgewählt.
Der Rapidly Exploring Random Tree (RRT) ist ein häufig verwendeter Suchbaum in diesem Kontext. Hier wird zunächst wie in 3. eine neue Konfiguration zufällig gewählt und ab dem nächsten Knoten im Graphen eine Schrittweite in Richtung dieser Konfiguration gegangen. Um einen Bias gegenüber der Zielkonfiguration zu erzeugen, kann zudem mit einer gewissen Wahrscheinlichkeit der zufällig gewählte Knoten mit dem tatsächlichen Zielknoten ersetzt werden. Bei RRT* wird bei der Erweiterung zusätzlich auf andere Verbindungen des Graphens zum neu berechneten Knoten geprüft, was die Länge der kürzesten Verbindung zwischen Start und Ziel noch einmal minimiert. Die kürzeste Verbindung kann beispielsweise mithilfe von A* oder Dijkstra ermittelt werden.

Multi-Query

Bei Multi-Query Verfahren können bereits berechnete Kanten des aufgebauten Graphens wiederverwendet werden. Im Probabilistic-Roadmap-Algorithmus werden in einem ersten Schritt n zufällige Knotenpunkte erzeugt und dann mit ihren nächsten Nachbarn verbunden, insofern keine Kollision vorliegt. Im Anschluss werden Start- und Zielknoten ergänzt und ebenfalls Nachbarn als Kanten im Graphen hinzugefügt. Am Ende kann dann wieder mithilfe von A* oder Dijkstra ein kürzester Weg ermittelt werden.
??
Potentialfeldmethode / Gradientenverfahren Genetische Algorithmen

3.3 Praxisbezug

Kinematisch (Winkelbegrenzung) Dynamisch (Geschwindigkeit) Einbezug der Constraints in den Algorithmen Praxis: OMPL (MoveIt+ ROS / Copelliasim Plugin)
Pfadplanungsalgorithmen können dazu beitragen, komplexe Pfade in einem Konfigurationsraum zu beschreiben. ROS (Robot Operating System) und CopelliaSim bieten bereits Frameworks, die dieses Problem lösen.

Da in dieser Arbeit der Schwerpunkt auf der Selbstkollision bei der Pfadplanung liegen soll und die inverse Kinematik nahezu ohne Singularitäten direkt berechnet werden kann, kann ein deutlich vereinfachter Algorithmus verwendet werden. Das setzt allerdings voraus, dass der Kollisionsraum schon zu Beginn der Berechnung vollständig bekannt ist. Aus diesen Gründen wurde in der Implementierung eine Bibliothek für das Überprüfen von Selbstkollisionen angelegt, in der alle möglichen Transformationen des UR5 auf fünf Grad genau abgespeichert sind.
Im implementierten Algorithmus kann zunächst für jede Zwischentransformation, die in Abschnitt 3.2 beschrieben wurde, mithilfe der inversen Kinematik aus Abschnitt 2 eine Liste möglicher Zwischenkonfigurationen erstellt werden. Dabei werden Konfigurationen, bei denen eine Kollision besteht, entfernt und Singularitäten durch das Addieren eines kleinen Werts ϵ umgangen.
Anschließend wird ein Graph erstellt, dessen Knoten die Konfigurationen der Zwischentransformationen darstellen, die Kanten die Übergänge zwischen diesen Konfigurationen beschreiben und deren Kantenlängen die Summe der Winkeldifferenzen aller Konfigurationen benachbarter Zeitschritte ausweisen. Nachdem der Graph erstellt wurde, müssen alle Knoten, die nicht vom Startknoten erreichbar sind, mithilfe einer Breitensuche entfernt werden. Danach kann zum Beispiel mit Dijkstra der kürzeste Weg im Graph beschrieben werden, der in diesem Fall genau den Weg darstellt, bei dem sich die Gelenke am wenigsten bewegen müssen.

4 Fazit und Ausblick

Mit den erläuterten Methoden kann ein kollisionsfreier gerader Pfad zwischen der momentanen Position und Stellung des Roboters zu einem erreichbaren Zielpunkt berechnet werden. Die beobachteten Selbstkollisionen sind vermutlich die Folge der Verwendung einer numerischen Bestimmung der inversen Kinematik mithilfe der Jakobi-Matrix, da diese Berechnungsmethodik unkompliziert ist, aber nur eine der acht möglichen Lösungen berechnen kann. Zudem wird gezeigt, dass beim Berechnen aller möglichen Konfigurationen einer vorgegebenen Bahn der Konfigurationsraum so weit eingegrenzt werden kann, dass in kurzer Zeit eine vollständige Lösung mit Dijkstra ermittelt werden kann.
Im nächsten Schritt kann nun ein Übergang in die Praxis erfolgen. Hierzu muss das Programm in die Steuerung des Roboters integriert werden und in Zukunft die Pfadplanung des UR5 ersetzen. Es sollte zudem untersucht werden, ob die vorgestellten Methoden bereits in OMPL (ROS mit MoveIt oder CopelliaSim Plugin) existieren und ob die hier eingesetzte Kollisionsbibliothek für den UR5 besser mit einem Kollisionstest des verwendeten Frameworks ersetzt werden sollte.

Literatur

[1] John J. Craig. *Introduction to Robotics: Mechanics and Control*. Third edition, Indian subcontinent adaptation. Chennai: Pearson, 2009. ISBN: 978-81-317-1836-0 (S. 1, 3, 9).

[2] M. R. De Gier. „Control of a Robotic Arm: Application to on-Surface 3D-printing“. In: (2015) (S. 8).

[3] J. Denavit und R. S. Hartenberg. „A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices“. In: *Journal of Applied Mechanics* 22.2 (1. Juni 1955), S. 215–221. ISSN: 0021-8936, 1528-9036. DOI: 10.1115/1.4011045 (S. 1).

[4] Kelsey P. Hawkins. *Analytic Inverse Kinematics for the Universal Robots UR-5/UR-10 Arms*. Technical Report. Georgia Institute of Technology, 7. Dez. 2013 (S. 9).

[5] Jahobr. *Coordinate systems and Denavit-Hartenberg parameters*. 2007. URL: https://commons.wikimedia.org/wiki/File:Denavit-Hartenberg-Transformation.svg (besucht am 24. 11. 2022) (S. 3).

[6] Rasmus Andersen. „Kinematics of a UR5“. In: (2018) (S. 3, 5, 9–13).

[7] ROS.org. *Urdf/XML/Model*. URL: http://wiki.ros.org/urdf/XML/model (besucht am 02. 12. 2022) (S. 4).

[8] Universal Robots. *Universal Robots - DH Parameters for Calculations of Kinematics and Dynamics*. URL: https://www.universal-robots.com/articles/ur/application-installation/dh-parameters-for-calculations-of-kinematics-and-dynamics/ (besucht am 02. 12. 2022) (S. 4–6).

[9] Universal Robots. *UR5 Technical Specifications*. URL: https://www.universal-robots.com/media/50588/ur5_en.pdf (besucht am 09. 12. 2022) (S. 4, 15).

Erklärung

Die vorliegende Arbeit habe ich selbstständig ohne Benutzung anderer als der angegebenen Quellen angefertigt. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten Quellen entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer oder anderer Prüfungen noch nicht vorgelegt worden.

Augsburg, den 31. 03. 2023 Nik Julin Nowoczyn