TECHNISCHE UNIVERSITÄT WIEN

AUTOMATION & CONTROL INSTITUTE
INSTITUT FÜR AUTOMATISIERUNGS- & REGELUNGSTECHNIK

Absolut genaue Trajektorienfolgeregelung in der Robotik mithilfe eines 2D-Lasertriangulationssensors

DIPLOMARBEIT

Ausgeführt zum Zwecke der Erlangung des akademischen Grades eines

Diplom-Ingenieurs (Dipl.-Ing.)

unter der Leitung von

Univ.-Prof. Dr. techn. A. Kugi
Dr. techn. C. Hartl-Nesic

eingereicht an der

Technischen Universität Wien
Fakultät für Elektrotechnik und Informationstechnik
Institut für Automatisierungs- und Regelungstechnik

von
Franz Stübler
Matrikelnummer 01325715

Wien, im November 2021

Gruppe für komplexe dynamische Systeme
A-1040 Wien, Gußhausstr. 27–29, Internet: https://www.acin.tuwien.ac.at

Vorwort

Das letzte Jahr war in vielerlei Hinsicht unstet und mit viel Unsicherheit verbunden. Ich möchte mich an dieser Stelle bei all jenen bedanken, die mich während dieser Diplomarbeit unterstützt und dadurch zum Gelingen dieser Arbeit beigetragen haben.

Ich möchte mich besonders bei meinem Betreuer Dr. techn. Christian Hartl-Nesic bedanken, der mich während der gesamten Dauer der Arbeit mit seiner Expertise und seinem Fachwissen unterstützt hat. Er ermöglichte mir bei virtuellen sowie persönlichen Diskussionen und Gesprächen meine Arbeit in die richtige Richtung voranzutreiben. Auch für die hilfreichen Anmerkungen und Kommentare zu diesem Dokument möchte ich mich herzlich bedanken.

Speziell danke ich Institutsvorstand Univ.-Prof. Dr. techn. Andreas Kugi für die Möglichkeit diese spannende Arbeit an dem Institut für Automatisierungs- und Regelungstechnik zu schreiben.

Ein großer Dank geht an meine Eltern, Sissy und Walter, die mich während meiner gesamten Studienzeit unterstützt haben und es so mir ermöglicht haben diesen Weg zu beschreiten. Das Korrekturlesen dieser Arbeit bildet dabei nur die Spitze des Eisberges.

Ein ganz besonderer Dank geht an meine Partnerin Katharina, die mich während der gesamten Zeit immer ermutigt und motiviert hat. Ihre bedingungslose Liebe und Geduld navigierten mich sicher durch diese unstetigen Zeiten.

Wien, im November 2021

Abstract

Industrial robots are multi-functional mechanical systems which can be used for the processing of objects and freeform surfaces. For a successful execution of the process, the end effector has to be able to follow a given trajectory with absolute accuracy. Due to non-ideal properties of the robot, such as tolerances in the kinematics and compliance in the mechanical structure, there is an inherent inaccuracy in the end effector pose of the robot.

The aim of this research work is to demonstrate that, with the assistance of a stationary 2D-laser triangulation sensor, it is possible for an industrial robot to process a workpiece with absolute accuracy. The pose of the workpiece, which consists of position and orientation, is calculated from the sensor's measurement data and fed back into the control system.

In order to determine the pose of the workpiece, the Iterative-Closest-Point-Algorithm is utilized and improved w. r. t. several aspects. Improvements involve the use of the so-called Approximated-Point-to-Plane error metric instead of the usual Point-to-Point error metric, which exhibits a faster convergence of the algorithm. Furthermore, this reduces the nonlinear optimization problem to a linear optimization, allowing to apply efficient solvers. As a result of further simplification in the search for the nearest point, the required runtime of the algorithm can be reduced from 116 s to 23 ms, without degrading the achieved accuracy of the pose identification. The industrial robot is controlled in the cartesian space with the use of the inverse dynamics.

The result of this research is presented in the form of a simulated work sequence. A KUKA LBR iiwa 14 R820 industrial robot is simulated with an inaccurate model. The robot moves a workpiece along a given trajectory at a speed of 7 mm/s. Without pose identification, a position error of up to 14 mm occurs due to the introduced model deviations. By measuring the workpiece with the MICRO-EPSILON scanCONTROL 2600-100 2D-laser triangulation sensor and feeding the result of the pose identification back into the control system, the maximum position error can be reduced to 0.6 mm. The achieved runtime of the pose identification allows deployment in an industrial application.

Kurzzusammenfassung

Industrieroboter sind multifunktionale mechanische Systeme, welche für die Bearbeitung von Objekten und Freiformoberflächen eingesetzt werden können. Für einen erfolgreichen Prozessablauf muss im Allgemeinen der Endeffektor einer vorgegebenen Trajektorie absolut genau folgen. Aufgrund von nichtidealen Eigenschaften des Roboters, wie Toleranzen in der Kinematik und fehlender Steifigkeit im mechanischen Aufbau, entsteht eine immanente Ungenauigkeit in der Endeffektorpose des Roboters.

Das Ziel der vorliegenden Forschungsarbeit ist es, zu zeigen, dass mithilfe eines ortsfesten 2D-Lasertriangulationssensors eine absolut genaue Bearbeitung eines Werkstückes durch einen Industrieroboter möglich ist. Dafür wird aus den Messdaten des Sensors die Pose des Werkstückes, bestehend aus Position und Orientierung, berechnet und in die Regelung rückgeführt.

Für die Bestimmung der Pose des Werkstücks wird der Iterative-Closest-Point-Algorithmus verwendet und weiterentwickelt. In der Weiterentwicklung wird die sogenannte Approximierte-Punkt-zu-Ebenen-Fehlermetrik anstatt der üblichen Punkt-zu-Punkt-Fehlermetrik verwendet, welche eine schnellere Konvergenz aufweist. Weiters reduziert sich das nichtlineare Optimierungsproblem auf eine lineare Optimierung, wodurch effiziente Lösungsverfahren angewendet werden können. Mithilfe einer weiteren Vereinfachung bei der Suche nach dem nächstgelegenen Punkt, kann die benötigte Laufzeit des gesamten Algorithmus von 116 s auf 23 ms reduziert werden, ohne Einschränkung bei der erreichten Genauigkeit der Posenbestimmung. Die Regelung des Industrieroboters erfolgt im kartesischen Arbeitsraum mithilfe der inversen Dynamik.

Das Ergebnis dieser Arbeit wird in Form eines simulierten Arbeitsablaufes veranschaulicht. Es wird ein Industrieroboter KUKA LBR iiwa 14 R820 mit fehlerbehaftetem Modell simuliert, welcher ein Werkstück entlang einer vorgegebenen Trajektorie mit einer Geschwindigkeit von 7 mm/s führt. Ohne Posenbestimmung entsteht aufgrund der eingeführten Modellfehler eine translatorische Abweichung von bis zu 14 mm. Durch die Vermessung des Werkstückes mit dem 2D-Lasertriangulationssensor MICRO-EPSILON scanCONTROL 2600-100 und der Rückführung des Ergebnisses der Posenbestimmung in die Regelung, kann die maximale Abweichung auf 0,6 mm reduziert werden. Die erreichte Laufzeit der Posenbestimmung erlaubt einen Einsatz in einer realen Umgebung.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

Industrieroboter sind programmierbare, automatisch arbeitende mechanische Systeme, welche unter anderem zum Bearbeiten oder Kontrollieren von Freiformoberflächen zum Einsatz kommen. Typische Anwendungen reichen von der Fertigung [1], dem Schweißen [2, 3] über das Lackieren [4] und das Polieren [5] bis hin zur Qualitätskontrolle [6] und Vermessung [7].

Roboter haben immer eine immanente Ungenauigkeit in der Positionierung des Endeffektors. Diese entsteht aufgrund von verschiedenen Faktoren, wie z. B. der nicht idealen Steifigkeit des Roboters, Toleranzen in der Produktion der Bestandteile des Roboters oder ungenauer Kalibrierung der Sensoren [8]. Die Positioniergenauigkeit des Endeffektors ist ein wesentliches Merkmal des Industrieroboters und hat bei vielen Anwendungen direkten Einfluss auf die Qualität des Endergebnisses [9]. Für spezielle Anwendungen wie Qualitätskontrolle und Vermessung ist eine hohe Genauigkeit Voraussetzung für korrekte Ergebnisse [7, 10]. Um eine Trajektorie für den Industrieroboter offline, d. h. ohne Zugriff auf den physischen Roboter, zu programmieren und zu testen, ist eine genaue Simulation des Roboters notwendig. Eine hohe Übereinstimmung zwischen der Bewegung des realen Roboters und dem Modell ist Voraussetzung für eine akkurate Simulation [11, 12].

1.1 Genauigkeit

Die ISO-Norm 9283 [13] beschreibt verschiedene Leistungsmerkmale und entsprechende Testmethoden für Industrieroboter. Unter anderem werden Genauigkeit, Steifigkeit und Zeitanforderungen an den Roboter behandelt. Wichtige Kenngrößen sind in dieser Arbeit die translatorische Absolutgenauigkeit und Wiederholgenauigkeit, welche im Folgenden definiert werden.

Die Messung der Positioniergenauigkeit erfolgt durch Vorgeben einer Position in kartesischen Koordinaten $\mathbf{p}_\mathrm{c} \in \mathbb{R}^3$ und wiederholtes Anfahren dieser sowie N-maliges Messen der erreichten Position $\mathbf{p}_i \in \mathbb{R}^3\,,\ i = 1, 2, \ldots, N$. Die Absolutgenauigkeit (AG) ist definiert als

$$\mathrm{AG} = \|\bar{\mathbf{p}} - \mathbf{p}_\mathrm{c}\| \tag{1.1a}$$

$$\text{mit} \quad \bar{\mathbf{p}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{p}_i \tag{1.1b}$$

und entspricht dem Abstand zwischen vorgegebener und mittlerer gemessener Position $\bar{\mathbf{p}}$. Dies beschreibt wie genau ein Roboter eine Position erreicht, wenn diese als kartesische

Koordinaten vorgegeben wird. Die Wiederholgenauigkeit (WG) ergibt sich mit (1.1b) zu

$$\mathrm{WG} = \left\| \bar{l} + 3s \right\| \tag{1.2a}$$

$$\text{mit} \quad \bar{l} = \frac{1}{N} \sum_{i=1}^{N} l_i \tag{1.2b}$$

$$l_i = \|\bar{\mathbf{p}} - \mathbf{p}_i\| \tag{1.2c}$$

$$s = \sqrt{\frac{\sum_{i=1}^{N} \left(\bar{l} - l_i\right)^2}{N-1}} \tag{1.2d}$$

und beschreibt die statistische Verteilung der gemessenen Positionen untereinander mithilfe der Standardabweichung s, wenn ein Industrieroboter eine Position wiederholt anfährt.

Für die Messung der Absolutgenauigkeit ($\mathrm{AG_P}$) entlang eines vorgegeben Pfades $\boldsymbol{\pi}_\mathrm{c}(k) : I \subseteq \mathbb{R} \mapsto \mathbb{R}^3$ wird der Pfad N-mal abgefahren. Dabei wird an diskreten Punkten $j = 1, 2, \ldots, M$ entlang des Pfades die Positionen $\mathbf{p}_{i,j}\,,\ i = 1, 2, \ldots, N$ gemessen und die dazu vorgegebene Position $\mathbf{p}_{\mathrm{c},j}$ gespeichert. Die $\mathrm{AG_P}$ ist definiert als

$$\mathrm{AG_P} = \max_{j=1,2,\ldots,M} \|\bar{\mathbf{p}}_j - \mathbf{p}_{\mathrm{c},j}\| \tag{1.3a}$$

$$\text{mit} \quad \bar{\mathbf{p}}_j = \frac{1}{N} \sum_{i=1}^{N} \mathbf{p}_{i,j} \tag{1.3b}$$

und entspricht der maximalen Differenz zwischen $\bar{\mathbf{p}}_j$ und $\mathbf{p}_{\mathrm{c},j}$.

1.2 Literaturstudie

In den Arbeiten [11, 12, 14] wurde die Genauigkeit von Industrierobotern mit externen Sensoren gemessen. Płaczek et al. [11] untersuchte den Industrieroboter Kuka KR 16-2 mit 1610 mm Reichweite und 16 kg Nutzlast. Als Messpunkte wurden verschiedene Punkte im Arbeitsraum des Roboters angefahren. Der maximale Absolutfehler war 2,21 mm und der maximale Wiederholungsfehler betrug 0,15 mm. Arbeit [12] analysierte die Genauigkeiten des Industrieroboters Kuka KR5 arc HW mit 1423 mm Reichweite und 5 kg Nutzlast. Die Wiederholgenauigkeit von fünf Positionen war im Mittel 0,07 mm mit einem Maximum von 0,12 mm und einer Absolutgenauigkeit von im Mittel 1,6 mm mit einem Maximum von 2,8 mm. Die Pfadgenauigkeit war bei 10 % der Maximalgeschwindigkeit 1,8 mm und 4,5 mm bei maximaler Geschwindigkeit. Slamani et al. [14] testeten die Genauigkeit des Industrieroboters Abb IRB 1600 mit 1450 mm Reichweite und 6 kg Nutzlast. Es wurde gezeigt, dass der maximal gemessene Absolutfehler 0,65 mm betrug und die maximale Wiederholgenauigkeit bei 37 µm lag. Es konnte auch gezeigt werden, dass bei längerem Betrieb des Industrieroboters ein zusätzlicher Fehler von 0,1 mm über die Zeit auftritt. Dieser entsteht durch die Erwärmung der Motoren und des ganzen Roboters unter Last.

Arbeit [8] teilt die Ursache der Ungenauigkeit von Industrieroboter in vier Fehlerkategorien ein, nämlich geometrische, dynamische, thermische und systematische Fehler. Durch Toleranzen der Bauteile des Roboters kommt es zu geometrischen Fehlern, welche im

Allgemeinen in der kinematischen Berechnung des Roboters nicht berücksichtigt werden. Dynamische Belastungen wie Lastmassen und Resonanzen führen zu zusätzlichen Fehlern während der Bewegung. Die thermische Ausdehnung der Bestandteile des Roboters führen zu Geometrieänderungen. Mögliche Wärmequellen sind z. B. Motoren, Getriebe und Lager. Systematische Eigenschaften wie z. B. eine fehlerhafte Kalibrierung oder Getriebespiel erzeugen Fehler welche nur bedingt korrigiert werden können.

Qin et al. [9] beschreiben wie die nicht ideale Steifigkeit eines Industrieroboters durch einen Zustandsbeobachter kompensiert werden kann. Dies wurde am Beispiel einer Rührreibschweißapplikation experimentell gezeigt. Bei dieser Art des Schweißens treten hohe Kräfte an dem Endeffektor auf, welche normalerweise durch die fehlende Steifigkeit im Getriebe des Roboters zu einem Positionsfehler führen würden, wodurch die Qualität der Schweißnaht sinkt. Dieser Fehler wird mit einem Zustandsbeobachter mithilfe der Motorströme geschätzt und durch den Regler kompensiert.

Die Optimierung der Absolutgenauigkeit eines Roboters an einzelnen Posen mithilfe eines optischen Tracking-Systems wurde in der Arbeit [15] beschrieben. Der Roboter wurde dafür zuerst in die Nähe der Zielpose gebracht. Die Differenz zwischen der Sollpose und der durch das optische Messsystem gemessenen Istpose wird als Regelfehler für einen PID-Regler verwendet. Der Ausgang des Reglers wird als Korrekturterm zur Zielpose an den Regler des Roboters weitergegeben. Dies wurde solange wiederholt bis der Fehler 50 µm unterschritten hat.

Arbeit [5] behandelt die Möglichkeit der Bearbeitung eines Werkstückes mit unbekannter Oberfläche durch einen Industrieroboter. Für die Bearbeitung ist die relative Entfernung und Orientierung zwischen der Objektoberfläche und dem Werkzeug, welches am Industrieroboter befestigt ist, von Bedeutung. Für die Bestimmung der Oberfläche wurde der Endeffektor mit zusätzlichen Punkt-Laserabstandssensoren ausgestattet. Mit diesen Messdaten wurde die Oberfläche lokal approximiert, wodurch der Industrieroboter das Werkzeug normal zur Objektoberfläche bewegen konnte.

Aristos et al. [16] beschäftigen sich mit der Posenbestimmung von einem Objekt im Arbeitsraum des Industrieroboters. Ein Lasertriangulationssensor ist am Endeffektor montiert und vermisst die Oberfläche des Objektes. Aus den Messdaten und der bekannten Oberfläche des Objektes wird die Pose relativ zum Industrieroboter bestimmt. Die Funktionsweise wurde an einzelnen statischen Positionen demonstriert. Der verwendete Algorithmus ist für moderat komplexe Objektoberflächen geeignet.

1.3 Ziel der Arbeit

Das Ziel dieser Arbeit ist es, ein Werkstück mithilfe eines Industrieroboters absolut genau entlang einer Trajektorie zu führen. Um die Absolutgenauigkeit zu erreichen, soll die Pose des Objektes, bestehend aus Position und Orientierung, von einem ortsfesten Sensor vermessen werden. Durch einen Algorithmus wird die Pose bestimmt und diese in der Trajektorienregelung mitberücksichtigt.

Beispielhaft wird dafür die Bearbeitung der Oberfläche eines Werkstückes durch einen Industrieroboter KUKA LBR iiwa 14 R820 simuliert. Dieser Prozess soll einem Poliervorgang nachgebildet sein. Dafür muss das Werkzeug eine Trajektorie auf dem Werkstück

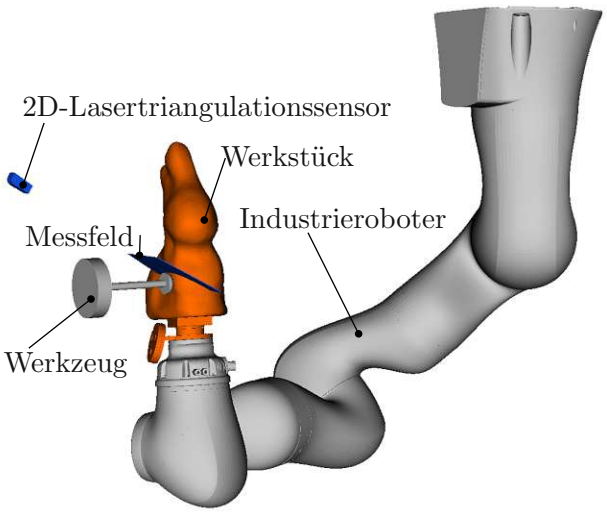


Abbildung 1.1: Aufbau der Simulationsumgebung mit Industrieroboter KUKA LBR iiwa 14 R820, 2D-Lasertriangulationssensor mit dazugehörigem Messfeld, Werkzeug und Werkstück.

abfahren und dabei immer normal zur Oberfläche orientiert sein. Für eine erfolgreiche Bearbeitung muss das Werkzeug im Kontakt mit der Oberfläche sein. In Abbildung 1.1 ist der Aufbau der Simulationsumgebung dargestellt. Der Roboter soll dabei mit einer für die Aufgabe unzureichenden Genauigkeit simuliert werden. Als Sensor soll ein 2D-Lasertriangulationssensor verwendet werden, welcher die Oberfläche des Werkstückes vermisst. Aus den Messdaten, der bekannten Oberfläche und der Pose des Sensors soll die genaue Pose des Werkstückes bestimmt werden. Diese Information soll in die Regelung in Echtzeit zurückgeführt und mit den Sensordaten des Roboters kombiniert werden, um die gewünschte Genauigkeit zu erreichen. In einer Parameterstudie soll die Auswirkung von verschiedenen Algorithmen, Ungenauigkeiten im Robotermodell und Effekte wie z. B. Rauschen auf die erreichte Gesamtgenauigkeit untersucht werden.

1.4 Gliederung der Arbeit

Die Arbeit ist in folgende Kapitel unterteilt. In Kapitel 2 wird die mathematische Beschreibung des Industrieroboters und des Lasertriangulationssensors hergeleitet. Dabei werden in Abschnitt 2.1 die benötigten Begriffe für die Arbeit definiert. Abschnitt 2.2 beschäftigt sich mit der mathematischen Herleitung des Industrieroboters und Abschnitt 2.3 mit der Funktionsweise sowie der Beschreibung des 2D-Lasertriangulationssensors.

Der Algorithmus zur Posenbestimmung des Werkstückes relativ zum Sensor wird in Kapitel 3 beschrieben. Die grundlegende Funktionsweise des Iterative-Closest-Point-Algorithmus (ICP-Algorithmus) wird in Abschnitt 3.1 erläutert. In Abschnitt 3.2 wird dieser Algorithmus angepasst, damit dieser für die Posenbestimmung des Werkstückes verwendet werden kann. Abschnitt 3.3 beschäftigt sich mit der Implementierung des Algorithmus und der Optimierung von dessen Laufzeit. In Abschnitt 3.4 wird die Performance verschiedener

ICP-Algorithmen verglichen.

In Kapitel 4 wird die Trajektorienplanung und die Regelung des Industrieroboters beschrieben. Die Planung des Pfades und der Trajektorie wird in Abschnitt 4.1 vorgestellt. Die Regelung mithilfe der inversen Dynamik und die Verwendung der zusätzlichen Information über die Pose wird in Abschnitt 4.2 erläutert.

Die Ergebnisse der Parameterstudie werden in Kapitel 5 präsentiert. Dafür werden in Abschnitt 5.1 der Aufbau der Simulation, die Einstellungen, Parameter und die verwendeten Variablen beschrieben. In Abschnitt 5.2 bis Abschnitt 5.6 werden die Ergebnisse präsentiert und erläutert. Abschließend werden diese in Abschnitt 5.7 diskutiert.

Zuletzt wird die Arbeit in Kapitel 6 zusammengefasst und es wird ein Ausblick auf weitere Forschungsmöglichkeiten gegeben.

2 Mathematische Modellierung

In diesem Kapitel wird die mathematische Beschreibung des Industrieroboters und des Sensors für die Simulation hergeleitet. Abschnitt 2.1 erläutert die in diesem Kapitel verwendeten Definitionen. In Abschnitt 2.2 wird das mathematische Modell des Industrieroboters KUKA LBR iiwa 14 R820 beschrieben. Die Modellierung des 2D-Lasertriangulationssensors MICRO-EPSILON scanCONTROL 2600-100 wird in Abschnitt 2.3 erläutert.

2.1 Definitionen

2.1.1 Geometrische Elemente

In diesem Abschnitt werden jene geometrischen Elemente eingeführt, welche in dieser Arbeit verwendet werden. Eine ausführliche Darstellung dieser Definitionen ist in [17] zu finden. Im weiteren Verlauf dieser Arbeit sind alle Vektoren und Punkte, sofern nicht anders bezeichnet, in einem dreidimensionalen kartesischen Koordinatensystem eingebettet. Die einzelnen Komponenten $i = x, y, z$ eines Vektors $\mathbf{v}^{\mathrm{T}} = \begin{bmatrix} a & b & c \end{bmatrix}$ werden mithilfe des Operators $(\mathbf{v})_i$, z. B.

$$(\mathbf{v})_x = a \tag{2.1}$$

ausgewählt. Strecken $L \subset \mathbb{R}^3$

$$L = \{\mathbf{l}_1, \mathbf{l}_2\} \tag{2.2}$$

sind über die zwei Endpunkte $\mathbf{l}_1$ und $\mathbf{l}_2$ beschrieben. Jeder Punkt $\mathbf{p}_L$ auf der Strecke ist durch die Parameterdarstellung

$$\mathbf{p}_L = \mathbf{l}_1 + t(\mathbf{l}_2 - \mathbf{l}_1)\,, \qquad t = [0, 1] \tag{2.3}$$

festgelegt. Dreiecke $D \subset \mathbb{R}^3$

$$D = \{\mathbf{d}_1, \mathbf{d}_2, \mathbf{d}_3\} \tag{2.4}$$

werden über die drei Eckpunkte des Dreiecks $\mathbf{d}_1$, $\mathbf{d}_2$ und $\mathbf{d}_3$ definiert. Jeder Punkt $\mathbf{p}_D$ innerhalb des Dreiecks $D \subset \mathbb{R}^3$ kann mithilfe sogenannter baryzentrischer Koordinaten in der Form

$$\mathbf{p}_D = a\mathbf{d}_1 + b\mathbf{d}_2 + c\mathbf{d}_3\,, \qquad a + b + c = 1 \wedge a, b, c \geq 0 \tag{2.5}$$

definiert werden. Die Bedingung $a + b + c = 1$ begrenzt $\mathbf{p}_D$ auf die Ebene, in welcher das Dreieck liegt und $a, b, c \geq 0$ begrenzt die Punkte auf die Fläche innerhalb des Dreiecks.

2.1.2 Homogene Transformation

In dieser Forschungsarbeit werden Koordinatentransformationen mithilfe homogener Transformation beschrieben, siehe [18]. Die geometrische Beschreibung des Koordinatensystems $\mathcal{Y}$ in Bezug auf das Koordinatensystem $\mathcal{X}$, dargestellt in $\mathcal{X}$, kann mithilfe der Verschiebung ${}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{d} \in \mathbb{R}^3$ und einer Rotationsmatrix ${}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R} \in \mathrm{SO}(3)$ angegeben werden. Die homogene Transformation

$$ {}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{H} = \begin{bmatrix} {}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R} & {}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix} \tag{2.6} $$

vereint beide Komponenten zu einer Matrix ${}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{H} \in \mathrm{SE}(3)$. Das Symbol $\mathbf{0}$ beschreibt in dieser Arbeit eine Nullmatrix entsprechender Dimension. Aus der Orthogonalität der Rotationsmatrix folgt für dessen Inverse

$$ \left({}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R}\right)^{-1} = \left({}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R}\right)^{\mathrm{T}} = {}^{\mathcal{X}}_{\mathcal{Y}}\mathbf{R} \,. \tag{2.7} $$

Die Inverse der homogenen Transformation ${}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{H}$ entspricht der geometrischen Darstellung des Koordinatensystems $\mathcal{X}$ im Bezug auf $\mathcal{Y}$, beschrieben in $\mathcal{Y}$, und wird mit ${}^{\mathcal{X}}_{\mathcal{Y}}\mathbf{H}$ bezeichnet. Mit der Eigenschaft (2.7) kann die Inverse von (2.6) in der Form

$$ \left({}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{H}\right)^{-1} = \begin{bmatrix} \left({}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R}\right)^{-1} & -\left({}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R}\right)^{-1}{}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{X}}_{\mathcal{Y}}\mathbf{R} & -{}^{\mathcal{X}}_{\mathcal{Y}}\mathbf{R}\,{}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix} = {}^{\mathcal{X}}_{\mathcal{Y}}\mathbf{H} \tag{2.8} $$

gebildet werden. Im weiteren Verlauf dieser Arbeit wird die homogene Transformation einer reinen Translation entlang einer Achse $i \in \{x, y, z\}$ um die Distanz d als $\mathbf{H}_{\mathrm{T}i,d}$ bezeichnet. Die homogene Transformation einer reinen Rotation um eine Achse $i \in \{x, y, z\}$ um den Winkel φ wird mit $\mathbf{H}_{\mathrm{R}i,\varphi}$ bezeichnet. Um die Rotationsmatrix $\mathbf{R}$ bzw. die Verschiebung $\mathbf{d}$ aus einer homogenen Transformation $\mathbf{H}$ auszuwählen, werden die Funktionen

$$ f_{\mathrm{R}}(\mathbf{H}) := \{\mathbf{R}\} \tag{2.9a} $$

$$ f_{\mathrm{d}}(\mathbf{H}) := \{\mathbf{d}\} \tag{2.9b} $$

definiert. Die Transformation einer Menge von Punkten $P = \{\mathbf{p}_1, \mathbf{p}_2, \ldots\}$ mit der homogenen Transformation $\mathbf{H}$ ist durch die Funktion

$$ f_{\mathrm{H}}(\mathbf{H}, P) := \{f_{\mathrm{R}}(\mathbf{H})\mathbf{p} + f_{\mathrm{d}}(\mathbf{H}) \mid \forall \mathbf{p} \in P\} \tag{2.10} $$

gegeben.

Der Zusammenhang der translatorischen Geschwindigkeit und der Winkelgeschwindigkeit zwischen drei sich zueinander bewegenden Koordinatensystemen $\mathcal{X}, \mathcal{Y}$ und $\mathcal{Z}$ lautet

$$ {}^{\mathcal{Z}}_{\mathcal{X}}\dot{\mathbf{d}} = {}^{\mathcal{Y}}_{\mathcal{X}}\dot{\mathbf{d}} + {}^{\mathcal{Y}}_{\mathcal{X}}\dot{\mathbf{R}}\,{}^{\mathcal{Z}}_{\mathcal{Y}}\mathbf{d} + {}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R}\,{}^{\mathcal{Z}}_{\mathcal{Y}}\dot{\mathbf{d}} \tag{2.11a} $$

$$ {}^{\mathcal{Z}}_{\mathcal{X}}\boldsymbol{\omega} = {}^{\mathcal{Y}}_{\mathcal{X}}\boldsymbol{\omega} + {}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{R}\,{}^{\mathcal{Z}}_{\mathcal{Y}}\boldsymbol{\omega} \,, \tag{2.11b} $$

siehe [19].

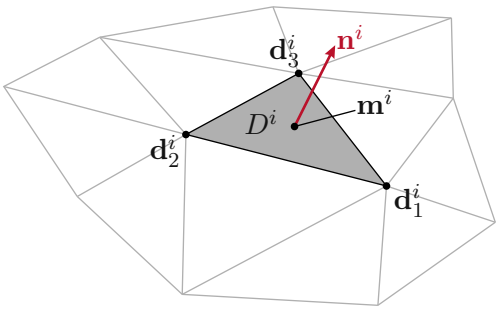


Abbildung 2.1: Diskrete Objektoberfläche mit dem Dreieck D^i, den Eckpunkten $\mathbf{d}_1^i, \mathbf{d}_2^i, \mathbf{d}_3^i$ und dem Normalvektor $\mathbf{n}^i$.

2.1.3 Diskrete Objektoberfläche

Die Oberfläche eines Objektes wird in dieser Arbeit durch eine Menge von Dreiecken

$$D = \left\{ D^1, D^2, \ldots, D^{N_\mathrm{D}} \right\} \tag{2.12a}$$

$$D^i = \left\{ \mathbf{d}_1^i, \mathbf{d}_2^i, \mathbf{d}_3^i \right\} \tag{2.12b}$$

mit $i = 1, 2, \ldots, N_\mathrm{D}$ beschrieben, siehe Abbildung 2.1. N_D bezeichnet dabei die Anzahl der Dreiecke. Die einzelnen Dreiecke D^i sind durch die drei Eckpunkte $\mathbf{d}_1^i, \mathbf{d}_2^i, \mathbf{d}_3^i$ im Raum festgelegt. Über gemeinsame Kanten und Ecken sind die Dreiecke miteinander verbunden. Zu jedem Dreieck D^i ist der Normalvektor $\mathbf{n}^i$ und die Menge der Normalvektoren N mithilfe des Kreuzproduktes $\times$ in der Form

$$\mathbf{n}^i = \frac{(\mathbf{d}_3^i - \mathbf{d}_1^i) \times (\mathbf{d}_2^i - \mathbf{d}_1^i)}{\left\| (\mathbf{d}_3^i - \mathbf{d}_1^i) \times (\mathbf{d}_2^i - \mathbf{d}_1^i) \right\|} \tag{2.13a}$$

$$N = \left\{ \mathbf{n}^1, \mathbf{n}^2, \ldots, \mathbf{n}^{N_\mathrm{D}} \right\} \tag{2.13b}$$

definiert. Die Normalvektoren sind in Richtung der Außenseite orientiert. Zusätzlich sind auch der Mittelpunkt $\mathbf{m}^i$ des Dreiecks D^i sowie die Menge der Mittelpunkte M durch

$$\mathbf{m}^i = \frac{\mathbf{d}_1^i + \mathbf{d}_2^i + \mathbf{d}_3^i}{3} \tag{2.14a}$$

$$M = \left\{ \mathbf{m}^1, \mathbf{m}^2, \ldots, \mathbf{m}^{N_\mathrm{D}} \right\} \tag{2.14b}$$

definiert.

2.2 Industrieroboter

In diesem Abschnitt wird das mathematische Modell des Industrieroboters KUKA LBR iiwa 14 R820 hergeleitet. Die Beschreibung der direkten Kinematik erfolgt mithilfe der homogenen Transformationen in Abschnitt 2.2.1. Im Abschnitt 2.2.2 wird die geometrische Manipulator Jacobi-Matrix aus der direkten Kinematik berechnet. Abschnitt 2.2.3 führt

die Dynamik des Roboters in Form der Bewegungsgleichung in Matrixform ein. Weiters werden in Abschnitt 2.2.4 zusätzliche Modellfehler eingeführt, welche das nominelle Modell in das reale Modell überführen.

2.2.1 Direkte Kinematik

Der Industrieroboter KUKA LBR iiwa 14 R820 ist ein serieller 7-Achs-Industrieroboter. Die Berechnung der Endeffektorpose in Abhängigkeit der Geometrie und der Gelenke $\mathbf{q}^{\mathrm{T}} = \begin{bmatrix} q_1 & q_2 & \cdots & q_7 \end{bmatrix}$ wird als direkte Kinematik bezeichnet, siehe [18]. Die Pose eines Objektes beschreibt in dieser Arbeit die Position und die Orientierung des Objektes im dreidimensionalen Raum. In dieser Arbeit wird die Endeffektorpose, welche sich aufgrund der nominellen Parameter ergibt, als nominelle Endeffektorpose bezeichnet.

Jedem Glied $i = 0, 1, \ldots, 7$ des Roboters wird ein entsprechendes Koordinatensystem $\mathcal{L}_i$ körperfest zugewiesen, siehe Abbildung 2.2. Das Koordinatensystem $\mathcal{L}_0$ entspricht dem Basiskoordinatensystem $\mathcal{B}$ und liegt an der Basis des Roboters. Der Ursprung des Koordinatensystems $\mathcal{L}_i$ liegt im Mittelpunkt der Achse zwischen dem Glied $i-1$ und dem Glied i. Die nominelle Pose des Endeffektorkoordinatensystem $\mathcal{E}$ liegt am Flansch des Roboters, siehe Abbildung 2.2. Das Weltkoordinatensystem $\mathcal{W}$ entspricht dem Referenzkoordinatensystem zu dem die anderen Koordinatensysteme bezogen sind. Im weiteren Verlauf dieser Arbeit werden aus Gründen der Lesbarkeit die einzelnen Achsen der Koordinatensysteme in den Abbildungen nicht gesondert ausgewiesen. Die x-, y- und z-Achsen werden in den Farben rot, grün und blau dargestellt. Die Transformation zwischen den Koordinatensystemen $\mathcal{L}_{i-1}$ zu $\mathcal{L}_i$ erfolgt mit der homogenen Transformation ${}^{\mathcal{L}_i}_{\mathcal{L}_{i-1}}\mathbf{H}$, welche durch

$$ {}^{\mathcal{L}_i}_{\mathcal{L}_{i-1}}\mathbf{H}(q_i) = \mathbf{H}_{\mathrm{Ty},d_{i,\mathrm{y}}}\, \mathbf{H}_{\mathrm{Tz},d_{i,\mathrm{z}}}\, \mathbf{H}_{\mathrm{Rx},\alpha_i}\, \mathbf{H}_{\mathrm{Rz},q_i}(q_i) \tag{2.15} $$

definiert ist. Die direkte Kinematik der nominellen Endeffektorpose $\mathcal{E}$ im Bezug auf das Basiskoordinatensystem $\mathcal{B}$ kann unter der Verwendung von (2.15) und den Gelenkwinkeln $\mathbf{q}$, gemäß

$$ {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{H}(\mathbf{q}) = {}^{\mathcal{L}_1}_{\mathcal{B}}\mathbf{H}\, {}^{\mathcal{L}_2}_{\mathcal{L}_1}\mathbf{H}(q_2)\, {}^{\mathcal{L}_3}_{\mathcal{L}_2}\mathbf{H}(q_3) \cdots {}^{\mathcal{L}_7}_{\mathcal{L}_6}\mathbf{H}(q_7)\, {}^{\mathcal{E}}_{\mathcal{L}_7}\mathbf{H} \tag{2.16} $$

berechnet werden. Die Parameter für (2.16) sind in Tabelle 2.1 aufgelistet und wurden [20, 21] entnommen. Eine Pose im dreidimensionalen Raum ist durch drei translatorischen und drei rotatorische Freiheitsgrade bestimmt. Zur einfachen Beschreibung der geometrischen Zusammenhänge wird in dieser Arbeit für die Pose die nicht-minimale Darstellung der homogenen Transformationen verwendet und bei Bedarf entsprechend umgewandelt.

2.2.2 Geometrische Manipulator Jacobi-Matrix

Die geometrische Manipulator Jacobi-Matrix ${}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}$, siehe [18], beschreibt den linearen Zusammenhang zwischen der Gelenkwinkelgeschwindigkeit $\dot{\mathbf{q}}$, der Lineargeschwindigkeit ${}^{\mathcal{E}}_{\mathcal{B}}\dot{\mathbf{d}}$ und der Winkelgeschwindigkeit ${}^{\mathcal{E}}_{\mathcal{B}}\boldsymbol{\omega}$ in der Form

$$ \begin{bmatrix} {}^{\mathcal{E}}_{\mathcal{B}}\dot{\mathbf{d}}(\mathbf{q}) \\ {}^{\mathcal{E}}_{\mathcal{B}}\boldsymbol{\omega}(\mathbf{q}) \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}_{\mathrm{v}}(\mathbf{q}) \\ {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}_{\omega}(\mathbf{q}) \end{bmatrix} \dot{\mathbf{q}} = {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \,. \tag{2.17} $$

Tabelle 2.1: Parameter des KUKA LBR iiwa 14 R820 für die direkte Kinematik, siehe [20, 21].

i	$d_{i,\mathrm{y}}$ mm	$d_{i,\mathrm{z}}$ mm	α_i °
1	0	152,5	0
2	−13	207,5	−90
3	−232,5	13	90
4	11	187,5	90
5	212,5	11	−90
6	−62	187,5	−90
7	−79,6	62	90
8	0	72,4	0

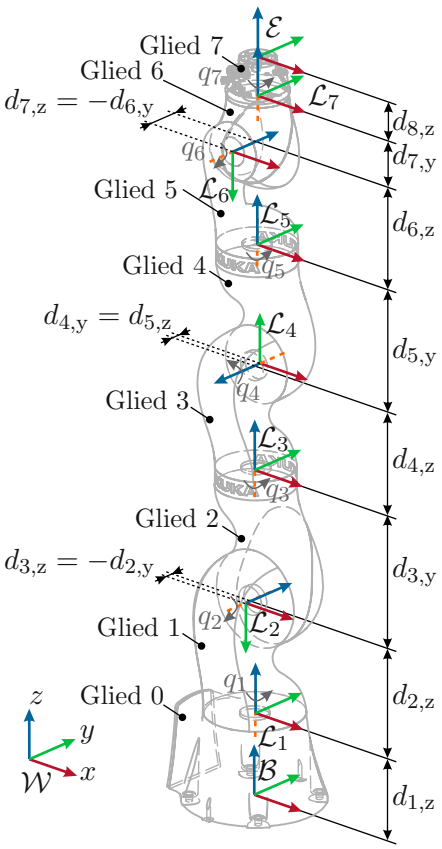


Abbildung 2.2: Wahl der Koordinatensysteme für den Industrieroboter KUKA LBR iiwa 14 R820 in der Stellung $\mathbf{q} = \mathbf{0}$.

Der translatorische Teil ${}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}_{\mathrm{v}}$ berechnet sich aus der Zeitableitung der Endeffektorposition ${}^{\mathcal{E}}_{\mathcal{B}}\mathbf{d} = f_{\mathrm{d}}\left({}^{\mathcal{E}}_{\mathcal{B}}\mathbf{H}\right)$ aus (2.16) zu

$$ {}^{\mathcal{E}}_{\mathcal{B}}\dot{\mathbf{d}}(\mathbf{q}) = \sum_{j=1}^{7} \frac{\partial\left({}^{\mathcal{E}}_{\mathcal{B}}\mathbf{d}(\mathbf{q})\right)}{\partial q_j}\dot{q}_j = {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}_{\mathrm{v}}(\mathbf{q})\dot{\mathbf{q}} \,. \tag{2.18} $$

Die schiefsymmetrische Matrix

$$ \dot{\mathbf{R}}\mathbf{R}^{\mathrm{T}} = \begin{bmatrix} 0 & -(\boldsymbol{\omega})_z & (\boldsymbol{\omega})_y \\ (\boldsymbol{\omega})_z & 0 & -(\boldsymbol{\omega})_x \\ -(\boldsymbol{\omega})_y & (\boldsymbol{\omega})_x & 0 \end{bmatrix} \tag{2.19} $$

beschreibt den Zusammenhang zwischen den einzelnen Komponenten des Vektors der Drehwinkelgeschwindigkeit $\boldsymbol{\omega}$ und der zeitlich veränderlichen Rotationsmatrix $\mathbf{R}(t)$. Die Funktion

$$ f_{\mathrm{S}}(\boldsymbol{\omega}) := \begin{bmatrix} 0 & -(\boldsymbol{\omega})_z & (\boldsymbol{\omega})_y \\ (\boldsymbol{\omega})_z & 0 & -(\boldsymbol{\omega})_x \\ -(\boldsymbol{\omega})_y & (\boldsymbol{\omega})_x & 0 \end{bmatrix} \tag{2.20} $$

bildet aus dem Vektor $\boldsymbol{\omega}$ die schiefsymmetrische Matrix und

$$ f_{\mathrm{US}}(f_{\mathrm{S}}(\boldsymbol{\omega})) := \boldsymbol{\omega} \tag{2.21} $$

ist die entsprechend inverse Funktion, welche aus der schiefsymmetrischen Matrix $\dot{\mathbf{R}}\mathbf{R}^{\mathrm{T}}$ den Vektor $\boldsymbol{\omega}$ bildet. Die Berechnung der Winkelgeschwindigkeit ${}^{\mathcal{E}}_{\mathcal{B}}\boldsymbol{\omega}$ aus ${}^{\mathcal{E}}_{\mathcal{B}}\mathbf{R} = f_{\mathrm{R}}\left({}^{\mathcal{E}}_{\mathcal{B}}\mathbf{H}\right)$ aus (2.16) und dessen Zeitableitung ergibt sich zu

$$ f_{\mathrm{S}}\left({}^{\mathcal{E}}_{\mathcal{B}}\boldsymbol{\omega}\right) = {}^{\mathcal{E}}_{\mathcal{B}}\dot{\mathbf{R}}(\mathbf{q})\left({}^{\mathcal{E}}_{\mathcal{B}}\mathbf{R}(\mathbf{q})\right)^{\mathrm{T}} = \sum_{j=1}^{7} \frac{\partial\left({}^{\mathcal{E}}_{\mathcal{B}}\mathbf{R}(\mathbf{q})\right)}{\partial q_j}\left({}^{\mathcal{E}}_{\mathcal{B}}\mathbf{R}(\mathbf{q})\right)^{\mathrm{T}}\dot{q}_j \tag{2.22} $$

Durch den Vergleich von (2.22) mit (2.17) ergibt sich die Vorschrift für den rotatorischen Teil der geometrischen Jacobi-Matrix ${}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}_{\omega}(\mathbf{q})$ gemäß

$$ {}^{\mathcal{E}}_{\mathcal{B}}\boldsymbol{\omega} = f_{\mathrm{US}}\left(f_{\mathrm{S}}\left({}^{\mathcal{E}}_{\mathcal{B}}\boldsymbol{\omega}\right)\right) = {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{J}_{\omega}(\mathbf{q})\dot{\mathbf{q}} \,. \tag{2.23} $$

2.2.3 Dynamik

Für die Herleitung der Dynamik des Industrieroboters KUKA LBR iiwa 14 R820 wird auf die Arbeit [18] verwiesen. Die Herleitung der Dynamik basiert auf den Euler-Lagrange-Gleichungen, welche über die kinetische und potentielle Energie berechnet werden. Das Ergebnis ist die Bewegungsgleichung des Starrkörpersystems in Matrixform

$$ \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \boldsymbol{\tau} \,. \tag{2.24} $$

Die Gleichung (2.24) besteht aus der generalisierten Massenmatrix $\mathbf{M}(\mathbf{q})$, der Matrix $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ der Coriolis- und Zentrifugalkräfte und dem Vektor der Potentialkräfte $\mathbf{g}(\mathbf{q})$ in den generalisierten Koordinaten $\mathbf{q}$. Der Vektor $\boldsymbol{\tau}$ umfasst die externen generalisierten Kräfte, die Motordrehmomente und dissipativen Kräfte. Die Parameter des Roboters wurden in [20, 21] experimentell identifiziert.

2.2.4 Modellfehler

In diesem Abschnitt wird zum nominellen Modell des Roboters ein zusätzlicher Modellfehler eingeführt, welcher eine Abweichung zwischen dem realen und nominellen Modell hervorruft. In der Simulation basiert der Roboter auf dem realen Modell während der Rest des Regelkreises hingegen das nominelle Modell verwendet.

Die mathematische Beschreibung des Fehlers basiert auf den Fehlerkategorien von Greenway [8]. Es werden ein geometrischer und systematischer Fehler modelliert, welche auch Einfluss auf das dynamische Modell haben. Dafür werden die homogenen Transformationen ${}^{\mathcal{L}_i}_{\mathcal{L}_{i-1}}\mathbf{H}$ von (2.15) um die zwei zusätzlichen Terme $\mathbf{H}_{\mathrm{Ty},\tilde{d}_{i,\mathrm{y}}}$ und $\mathbf{H}_{\mathrm{Rz},\tilde{q}_i}$ gemäß

$$
{}^{\tilde{\mathcal{L}}_i}_{\tilde{\mathcal{L}}_{i-1}}\mathbf{H} = \mathbf{H}_{\mathrm{Ty},d_{i,\mathrm{y}}}\,\mathbf{H}_{\mathrm{Tz},d_{i,\mathrm{z}}}\,\mathbf{H}_{\mathrm{Ty},\tilde{d}_{i,\mathrm{y}}}\,\mathbf{H}_{\mathrm{Rx},\alpha_i}\,\mathbf{H}_{\mathrm{Rz},q_i}\,\mathbf{H}_{\mathrm{Rz},\tilde{q}_i} \tag{2.25}
$$

erweitert. Die realen Posen der Koordinatensysteme und die realen Parameter des Industrieroboters werden gegenüber den nominellen Varianten mit einer zusätzlichen Tilde $(\tilde{\ })$ gekennzeichnet. Die Transformation $\mathbf{H}_{\mathrm{Ty},\tilde{d}_{i,\mathrm{y}}}$ beschreibt einen geometrischen Fehler beim Glied i in Richtung der y-Achse. Der Term $\mathbf{H}_{\mathrm{Rz},\tilde{q}_i}$ fügt einen systematischen Fehler des Winkelsensors am Gelenk i hinzu, wodurch ein Unterschied zwischen dem gemessenen Winkel q_i und dem realen Winkel $q_i + \tilde{q}_i$ entsteht. Die Fehlerterme werden in den Fehlervektoren

$$
\tilde{\mathbf{d}}_{\mathrm{y}} = \begin{bmatrix} \tilde{d}_{1,\mathrm{y}} & \tilde{d}_{2,\mathrm{y}} & \cdots & \tilde{d}_{7,\mathrm{y}} \end{bmatrix}^{\mathrm{T}} \tag{2.26a}
$$

$$
\tilde{\mathbf{q}} = \begin{bmatrix} \tilde{q}_1 & \tilde{q}_2 & \cdots & \tilde{q}_7 \end{bmatrix}^{\mathrm{T}} \tag{2.26b}
$$

zusammengefasst. In dieser Arbeit werden die Fehlerterme als zeitunabhängig angenommen, d. h. es gilt

$$
\dot{\tilde{\mathbf{d}}}_{\mathrm{y}} = \mathbf{0} \tag{2.27a}
$$

$$
\dot{\tilde{\mathbf{q}}} = \mathbf{0}\ . \tag{2.27b}
$$

Durch Ersetzen von (2.25) in (2.16) ergibt sich die reale direkte Kinematik in der Form

$$
{}^{\tilde{\mathcal{E}}}_{\mathcal{B}}\mathbf{H}\left(\mathbf{q}, \tilde{\mathbf{d}}_{\mathrm{y}}, \tilde{\mathbf{q}}\right) = \begin{bmatrix} {}^{\tilde{\mathcal{E}}}_{\mathcal{B}}\mathbf{R}\left(\mathbf{q}, \tilde{\mathbf{d}}_{\mathrm{y}}, \tilde{\mathbf{q}}\right) & {}^{\tilde{\mathcal{E}}}_{\mathcal{B}}\mathbf{d}\left(\mathbf{q}, \tilde{\mathbf{d}}_{\mathrm{y}}, \tilde{\mathbf{q}}\right) \\ \mathbf{0} & 1 \end{bmatrix}\ . \tag{2.28}
$$

Der Modellfehler in (2.25) führt zu einem Unterschied zwischen der realen und der nominellen geometrischen Manipulator Jacobi-Matrix (2.17) sowie zwischen der realen und der nominellen Dynamik (2.24). Die reale geometrische Manipulator Jacobi-Matrix und die reale Dynamik lauten

$$
\begin{bmatrix} {}^{\tilde{\mathcal{E}}}_{\mathcal{B}}\dot{\mathbf{d}} \\ {}^{\tilde{\mathcal{E}}}_{\mathcal{B}}\boldsymbol{\omega} \end{bmatrix} = {}^{\tilde{\mathcal{E}}}_{\mathcal{B}}\mathbf{J}\left(\mathbf{q}, \tilde{\mathbf{d}}_{\mathrm{y}}, \tilde{\mathbf{q}}\right)\dot{\mathbf{q}} \tag{2.29a}
$$

$$
\tilde{\mathbf{M}}\left(\mathbf{q}, \tilde{\mathbf{d}}_{\mathrm{y}}, \tilde{\mathbf{q}}\right)\ddot{\mathbf{q}} + \tilde{\mathbf{C}}\left(\mathbf{q}, \tilde{\mathbf{d}}_{\mathrm{y}}, \tilde{\mathbf{q}}, \dot{\mathbf{q}}\right)\dot{\mathbf{q}} + \tilde{\mathbf{g}}\left(\mathbf{q}, \tilde{\mathbf{d}}_{\mathrm{y}}, \tilde{\mathbf{q}}\right) = \boldsymbol{\tau}\ . \tag{2.29b}
$$

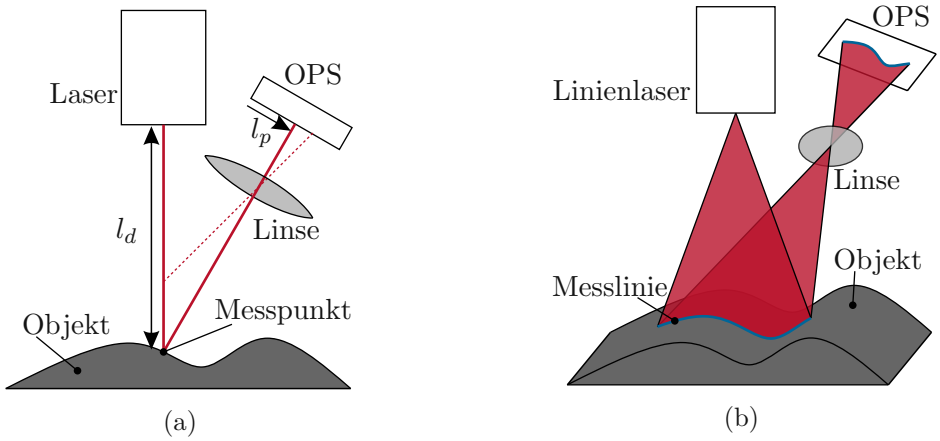


Abbildung 2.3: Darstellung der Funktionsweise des Lasertriangulationssensors mit Laser, Messpunkt, Linse und optischem Positionssensor (OPS), (a) 1D-Lasertriangulationssensor, (b) 2D-Lasertriangulationssensor.

2.3 2D-Lasertriangulationssensor

In diesem Abschnitt wird das mathematische Modell des 2D-Lasertriangulationssensors Micro-Epsilon scanCONTROL 2600-100 hergeleitet. Dafür wird in Abschnitt 2.3.1 die Funktionsweise des Sensors zuerst für eine punktförmige Messung erklärt und danach auf eine linienförmige Messung erweitert. In Abschnitt 2.3.2 werden die entsprechenden Modellannahmen und Parameter eingeführt und die Herleitung der mathematischen Modellierung erfolgt in Abschnitt 2.3.3. Im Anschluss wird in Abschnitt 2.3.4 ein zusätzlicher Messfehler eingeführt, um die Ungenauigkeiten des Sensors zu modellieren. Um die Simulationsgeschwindigkeit zu erhöhen, wird in Abschnitt 2.3.5 noch eine Möglichkeit beschrieben wie die Laufzeit des Algorithmus optimiert werden kann.

2.3.1 Funktionsweise

Die Lasertriangulation gehört zu den berührungslosen optischen Distanzmessverfahren. In Abbildung 2.3a ist der vereinfachte Aufbau für eine punktförmige Messung dargestellt. Der Laser, die Linse und der optische Linienpositionssensor (OPS) sind im Allgemeinen zusammen in einem Gehäuse untergebracht und stehen zueinander in einer festen geometrischen Beziehung. Der Laser, welcher auf das Messobjekt gerichtet ist, emittiert einen Laserstrahl. Dieser wird vom Messobjekt reflektiert und über eine Linse auf den OPS fokussiert. Aus der gemessenen Länge l_p auf dem OPS und der Geometrie des Aufbaus kann die Entfernung l_d berechnet werden. Für die Erweiterung der punktförmigen Messung auf eine Linienmessung wird der punktförmige Laserstrahl auf eine Linie aufgeweitet und der optische Linienpositionssensor durch einen Flächenpositionssensor ersetzt. Der Messaufbau ist in Abbildung 2.3b illustriert. Die Berechnung der Entfernung der einzelnen Messpunkte auf der Messlinie erfolgt nach dem selben Prinzip.

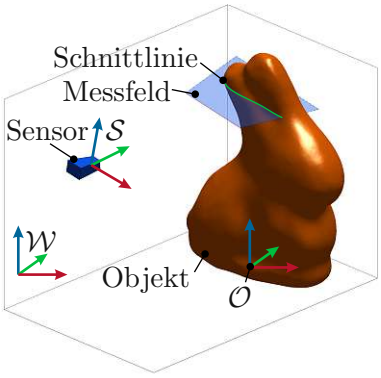


Abbildung 2.4: Messaufbau mit dem Objekt, dem 2D-Lasertriangulationssensor, dem Messfeld und der Schnittlinie.

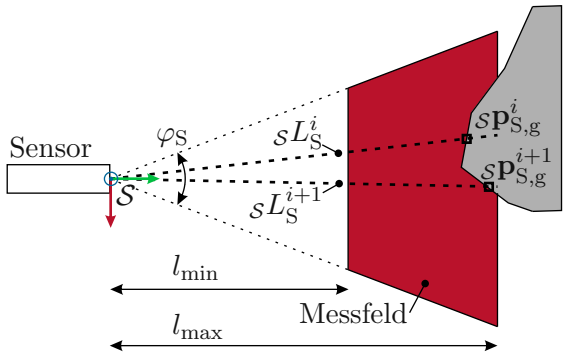


Abbildung 2.5: Schematische Darstellung einer Messung des 2D-Lasertriangulationssensors mit Öffnungswinkel φ_S, minimaler und maximaler Messdistanz l_{min} bzw. l_{max}, Messlinien ${}_{\mathcal{S}}L_S^i$ und gültige Messpunkten ${}_{\mathcal{S}}\mathbf{p}_{S,g}^i$.

2.3.2 Modellannahmen und Parameter

Das Ziel der Modellierung ist es, den Messvorgang durch den 2D-Lasertriangulationssensor mathematisch zu beschreiben. In Abbildung 2.4 ist der Sensor mit dem Objekt im Messfeld dargestellt. Der 2D-Lasertriangulationssensor vermisst die Oberfläche des zu bearbeitenden Objektes, welches im Objektkoordinatensystem $\mathcal{O}$ beschrieben ist. Das Sensorkoordinatensystem $\mathcal{S}$ beschreibt die Pose des Sensors im Raum. Das Koordinatensystem des Sensors $\mathcal{S}$ und des Objektes $\mathcal{O}$ werden auf das Weltkoordinatensystem $\mathcal{W}$ bezogen.

Eine schematische Darstellung der modellierten Messung mit den Parametern ist in Abbildung 2.5 veranschaulicht. Für die Modellierung werden folgende Annahmen getroffen:

- Die Messwerte werden auf das Koordinatensystem $\mathcal{S}$ bezogen.
- Die y-Achse des Sensors ist in Richtung des Messfeldes orientiert.
- Das Messfeld liegt in der xy-Ebene von $\mathcal{S}$.

- Die Oberfläche des Objektes wird durch eine diskrete Oberfläche dargestellt, siehe Abschnitt 2.1.3.
- Das Messfeld des Sensors hat einen Öffnungswinkel von φ_S.
- Das Messfeld ist durch den minimalen Abstand l_{min} und den maximalen Abstand l_{max} begrenzt.
- Das Messfeld wird durch eine Anzahl N_S diskreter Messlinien ${}_S L_S^i$, $i = 1, 2, \ldots, N_S$ modelliert.
- Wenn ein Schnittpunkt außerhalb des Messfeldes liegt ist dieser ungültig.
- Die Menge aller gültigen Messpunkte ${}_S P_{S,g}$ sind die exakten Schnittpunkte zwischen der diskreten Oberfläche und den Messlinien.
- Das Rauschen des Sensors wird durch eine Normalverteilung simuliert und die Menge der entsprechenden Messpunkte mit Rauschen werden mit ${}_S P_S$ bezeichnet.
- Der Sensor hat eine maximale Abtastfrequenz von $\frac{1}{T_S}$.

In der Modellierung wird der Pfad des Lasers vom Objekt zum OPS vernachlässigt, vgl. Abbildung 2.3b. Dadurch kann es beim realen Sensor zu Verschattungen kommen, welche beim modellierten Sensor nicht auftreten würden. Der OPS und der Linienlaser sind im scanCONTROL 2600-100 örtlich nahe untergebracht, wodurch der Effekt der Verschattung minimiert wird. Weiters wird angenommen, dass bei dem gewählten Objekt aufgrund der Form wenig Verschattung auftritt. Die Parameter des 2D-Lasertriangulationssensors Micro-Epsilon scanCONTROL 2600-100 werden aus dem Datenblatt [22] entnommen und sind in Tabelle 2.2 zusammengefasst.

2.3.3 Berechnung der gültigen Messpunkte

Für die mathematische Beschreibung müssen die Schnittpunkte zwischen den Messlinien und der diskreten Oberfläche berechnet werden. Dafür wird zunächst beschrieben, wie

Tabelle 2.2: Parameter des 2D-Lasertriangulationssensors Micro-Epsilon scanCONTROL 2600-100 aus dem Datenblatt [22].

Symbol	Einheit	Wert
φ_S	°	21,4
N_S	1	640
l_{min}	mm	190
l_{max}	mm	290
T_S	ms	$0,\overline{3}$
σ_E	µm	65
μ_E	m	0

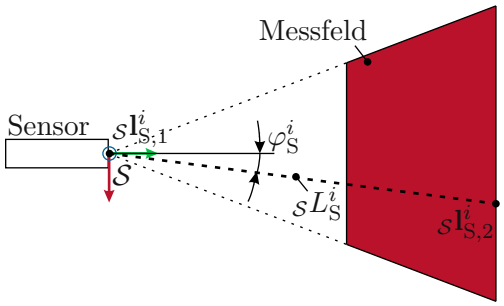


Abbildung 2.6: Darstellung einer Messlinie ${}_{\mathcal{S}}L^i_{\mathrm{S}}$ mit Winkel φ^i_{S}, Endpunkten ${}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},1}$ und ${}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},2}$.

ein Schnittpunkt zwischen einer Messlinie ${}_{\mathcal{S}}L^i_{\mathrm{S}}$ und dem Dreieck ${}_{\mathcal{O}}D^j_{\mathrm{O}}$ berechnet und auf Gültigkeit geprüft wird. Dies wird auf alle Dreiecke ${}_{\mathcal{O}}D_{\mathrm{O}}$ erweitert und der am nächsten liegende Schnittpunkt ist der gültige Messpunkt ${}_{\mathcal{S}}\mathbf{p}^i_{\mathrm{S,g}}$. Durch Wiederholen des Vorgangs für alle Messlinien ${}_{\mathcal{S}}L_{\mathrm{S}}$ ergibt sich die mathematische Beschreibung des 2D-Lasertriangulationssensors ohne Messrauschen.

Die diskrete Oberfläche besteht aus N_{O} Dreiecken welche in der Menge

$$ {}_{\mathcal{O}}D_{\mathrm{O}} = \left\{ {}_{\mathcal{O}}D^1_{\mathrm{O}},\, {}_{\mathcal{O}}D^2_{\mathrm{O}}, \ldots,\, {}_{\mathcal{O}}D^{N_{\mathrm{O}}}_{\mathrm{O}} \right\} \tag{2.30a} $$

$$ {}_{\mathcal{O}}D^j_{\mathrm{O}} = \left\{ {}_{\mathcal{O}}\mathbf{d}^j_{\mathrm{O},1},\, {}_{\mathcal{O}}\mathbf{d}^j_{\mathrm{O},2},\, {}_{\mathcal{O}}\mathbf{d}^j_{\mathrm{O},3} \right\} \tag{2.30b} $$

$$ j = 1, 2, \ldots, N_{\mathrm{O}} \tag{2.30c} $$

relativ zum Objektkoordinatensystem $\mathcal{O}$ zusammengefasst sind, siehe Abschnitt 2.1.3.

Die Endpunkte einer Messlinie ${}_{\mathcal{S}}L^i_{\mathrm{S}}$ ergeben sich aus Abbildung 2.6 mit den Parametern aus Tabelle 2.2 in Abhängigkeit des Winkels φ^i_{S} in der Form

$$ {}_{\mathcal{S}}L^i_{\mathrm{S}} = \left\{ {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},1},\, {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},2} \right\} \tag{2.31a} $$

$$ \varphi^i_{\mathrm{S}} = \frac{\varphi_{\mathrm{S}}}{N_{\mathrm{S}} - 1}(i-1) - \frac{\varphi_{\mathrm{S}}}{2} \tag{2.31b} $$

$$ {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},1} = \mathbf{0} \tag{2.31c} $$

$$ {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},2} = \begin{bmatrix} l_{\max} \tan \varphi^i_{\mathrm{S}} \\ l_{\max} \\ 0 \end{bmatrix} \tag{2.31d} $$

mit $i = 1, 2, \ldots, N_{\mathrm{S}}$. Die Punkte entlang der Messlinie werden durch die Parameterdarstellung (2.3) beschrieben

$$ {}_{\mathcal{S}}\mathbf{p}^i_{\mathrm{L}} = {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},1} + t^i_{\mathrm{S}}\left({}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},2} - {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},1} \right) \tag{2.32} $$

Der Parameterbereich des gültigen Messfeldes, vgl. Abbildung 2.5, ist für alle Messlinien gleich und beträgt

$$ t^i_{\mathrm{S}} = [t_{\mathrm{G}}, 1] \tag{2.33a} $$

$$ \text{mit} \quad t_{\mathrm{G}} = \frac{l_{\min}}{l_{\max}} \,. \tag{2.33b} $$

Ein Objekt zwischen Sensor und Messfeld würde zu einer Verschattung des Messfeldes führen. Um dies zu erkennen, wird der komplette Parameterbereich

$$t_{\mathrm{S}}^{i} = [0, 1] \tag{2.34}$$

benötigt. Die Messlinien werden in der Menge

$${}_{\mathcal{S}}L_{\mathrm{S}} = \left\{ {}_{\mathcal{S}}L_{\mathrm{S}}^{1}, {}_{\mathcal{S}}L_{\mathrm{S}}^{2}, \ldots, {}_{\mathcal{S}}L_{\mathrm{S}}^{N_{\mathrm{S}}} \right\} \tag{2.35}$$

relativ zum Sensorkoordinatensystem $\mathcal{S}$ zusammengefasst.

Im nächsten Schritt sollen die Messwerte des Sensors relativ zum Sensorkoordinatensystem $\mathcal{S}$ berechnet werden. Dazu wird im Folgenden der Schnittpunkt einer Messlinie ${}_{\mathcal{S}}L_{\mathrm{S}}^{i}$ mit dem Dreieck ${}_{\mathcal{O}}D_{\mathrm{O}}^{j}$ betrachtet. Die Eckpunkte des Dreiecks ${}_{\mathcal{O}}D_{\mathrm{O}}^{j}$ sind relativ zum Objektkoordinatensystem $\mathcal{O}$ definiert. Für die Berechnung des Schnittpunktes müssen diese auf das Sensorkoordinatensystem $\mathcal{S}$ bezogen werden. Über das Weltkoordinatensystem $\mathcal{W}$ kann die homogene Transformation ${}_{\mathcal{S}}^{\mathcal{O}}\mathbf{H}$ durch

$${}_{\mathcal{S}}^{\mathcal{O}}\mathbf{H} = \left({}_{\mathcal{W}}^{\mathcal{S}}\mathbf{H} \right)^{-1} {}_{\mathcal{W}}^{\mathcal{O}}\mathbf{H} \tag{2.36}$$

berechnet werden. Die Berechnung der transformierten Eckpunkte erfolgt mit (2.36) und der Funktion (2.10) in der Form

$${}_{\mathcal{S}}D_{\mathrm{O}}^{j} = f_{\mathrm{H}}\left({}_{\mathcal{S}}^{\mathcal{O}}\mathbf{H}, {}_{\mathcal{O}}D_{\mathrm{O}}^{j} \right) . \tag{2.37}$$

Die Beschreibung der Punkte ${}_{\mathcal{S}}\mathbf{p}_{\mathrm{D}}^{j}$ innerhalb des Dreiecks ${}_{\mathcal{S}}D_{\mathrm{O}}^{j}$ erfolgt mithilfe der baryzentrischen Koordinaten, siehe (2.5). Wenn eine Linie und eine Ebene nicht exakt parallel sind, existiert stets ein Schnittpunkt zwischen diesen und kann berechnet werden, indem die Parameterdarstellung der Linie (2.3) und des Dreiecks (2.5) gleichgesetzt werden. Daraus ergibt sich ein Gleichungssystem mit den gesuchten Parameter $t_{\mathrm{S}}^{i,j}$, $a_{\mathrm{O}}^{i,j}$, $b_{\mathrm{O}}^{i,j}$ und $c_{\mathrm{O}}^{i,j}$ des Schnittpunktes gemäß

$${}_{\mathcal{S}}\mathbf{l}_{\mathrm{S},1}^{i} + t_{\mathrm{S}}^{i,j}\left({}_{\mathcal{S}}\mathbf{l}_{\mathrm{S},2}^{i} - {}_{\mathcal{S}}\mathbf{l}_{\mathrm{S},1}^{i} \right) = a_{\mathrm{O}}^{i,j}\, {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},1}^{j} + b_{\mathrm{O}}^{i,j}\, {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},2}^{j} + c_{\mathrm{O}}^{i,j}\, {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},3}^{j} \tag{2.38a}$$

$$c_{\mathrm{O}}^{i,j} = 1 - a_{\mathrm{O}}^{i,j} - b_{\mathrm{O}}^{i,j} . \tag{2.38b}$$

$a_{\mathrm{O}}^{i,j}$, $b_{\mathrm{O}}^{i,j}$ und $c_{\mathrm{O}}^{i,j}$ sind die baryzentrischen Koordinaten und $t_{\mathrm{S}}^{i,j}$ der Parameter der Strecke. Durch das Umformen von (2.38b) nach $c_{\mathrm{O}}^{i,j}$ und Einsetzen in (2.38a) ergibt sich das lineare Gleichungssystem mit drei Unbekannten zu

$$\mathbf{A}\mathbf{x}^{i,j} = \mathbf{b} \tag{2.39a}$$

$$\text{mit} \quad \mathbf{A} = \begin{bmatrix} {}_{\mathcal{S}}\mathbf{l}_{\mathrm{S},1}^{i} - {}_{\mathcal{S}}\mathbf{l}_{\mathrm{S},2}^{i} & {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},1}^{j} - {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},3}^{j} & {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},2}^{j} - {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},3}^{j} \end{bmatrix} \tag{2.39b}$$

$$\mathbf{x}^{i,j} = \begin{bmatrix} t_{\mathrm{S}}^{i,j} & a_{\mathrm{O}}^{i,j} & b_{\mathrm{O}}^{i,j} \end{bmatrix}^{\mathrm{T}} \tag{2.39c}$$

$$\mathbf{b} = {}_{\mathcal{S}}\mathbf{l}_{\mathrm{S},1}^{i} - {}_{\mathcal{S}}\mathbf{d}_{\mathrm{O},3}^{j} . \tag{2.39d}$$

Der Lösungsvektor

$$\mathbf{x}^{i,j} = \mathbf{A}^{-1}\mathbf{b} \tag{2.40}$$

kann mithilfe der Inversen $\mathbf{A}^{-1}$ berechnet werden. Der Schnittpunkt zwischen Messlinie ${}_{\mathcal{S}}L^i_{\mathrm{S}}$ und dem Dreieck ${}_{\mathcal{S}}D^j_{\mathrm{O}}$ existiert genau dann, wenn der Lösungsvektor $\mathbf{x}^{i,j}$ die Bedingungen (2.34) und $a^{i,j}, b^{i,j}, c^{i,j} \geq 0$ erfüllt, vgl. (2.5). Dadurch sind auch die Schnittpunkte enthalten, welche auf der Messlinie ${}_{\mathcal{S}}L^i_{\mathrm{S}}$ liegen, aber nicht im gültigen Bereich (2.33) sind. Dadurch kann eine Verschattung des Messfeldes erkannt werden.

Die obigen Berechnungsschritte werden für alle Dreiecke ${}_{\mathcal{O}}D^j_{\mathrm{O}} \in {}_{\mathcal{O}}D_{\mathrm{O}}$ mit der Messlinie ${}_{\mathcal{S}}L^i_{\mathrm{S}}$ wiederholt. Die Indizes j werden in der Menge G^i für jede Messlinie zusammengefasst. Der vorläufige Schnittpunkt ist der zum Sensor am nächsten liegende Schnittpunkt. Der dazugehörige vorläufige Parameter $t^{i,\mathrm{v}}_{\mathrm{S}}$ entspricht dem minimalen Parameter der Strecke für diese Messlinie

$$t^{i,\mathrm{v}}_{\mathrm{S}} = \min_{j \in G^i} t^{i,j}_{\mathrm{S}} \ . \tag{2.41}$$

Dieser vorläufige Parameter ist gültig wenn die Bedingung des Messfeldes (2.33) erfüllt wird. Der korrekte gültige Schnittpunkt ergibt sich in der Form

$${}_{\mathcal{S}}\mathbf{p}^i_{\mathrm{S,g}} = \begin{cases} {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},1} + t^{i,\mathrm{v}}_{\mathrm{S}}\left({}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},2} - {}_{\mathcal{S}}\mathbf{l}^i_{\mathrm{S},1}\right), & \text{wenn} \quad t^{i,\mathrm{v}}_{\mathrm{S}} > t_{\mathrm{G}} \\ \text{ungültig} \ , & \text{sonst} \ . \end{cases} \tag{2.42}$$

Durch Wiederholen des gesamten Vorgangs für alle Messlinien ${}_{\mathcal{S}}L_{\mathrm{S}}$ ergibt sich schließlich die Menge ${}_{\mathcal{S}}P_{\mathrm{S,g}}$ in der alle gültigen Messpunkte ohne Messfehler zusammengefasst sind.

2.3.4 Messrauschen

Das Messrauschen des 2D-Lasertriangulationssensors Micro-Epsilon scanCONTROL 2600-100 kann laut Datenblatt [22] durch eine Normalverteilung $N(\mu_{\mathrm{E}}, \sigma_{\mathrm{E}})$ mit einem Erwartungswert $\mu_{\mathrm{E}} = 0\,\mathrm{m}$ und einer Standardabweichung $\sigma_{\mathrm{E}} = 65\,\mu\mathrm{m}$ abgeschätzt werden, siehe Tabelle 2.2. Der Startpunkt jeder Messlinie ist der Ursprung des Sensorkoordinatensystems $\mathcal{S}$, gemäß (2.31c), demnach ergibt sich der Messpunkt mit zufälligem Messfehler $x_{\mathrm{E}} \sim N(\mu_{\mathrm{E}}, \sigma_{\mathrm{E}})$ zu

$${}_{\mathcal{S}}\mathbf{p}^i_{\mathrm{S}} = {}_{\mathcal{S}}\mathbf{p}^i_{\mathrm{S,g}} + x_{\mathrm{E}} \frac{\mathbf{p}^i_{\mathrm{S,g}}}{\left\| {}_{\mathcal{S}}\mathbf{p}^i_{\mathrm{S,g}} \right\|} \ . \tag{2.43}$$

Die Messpunkte des 2D-Lasertriangulationssensors mit Messfehler werden in der Menge ${}_{\mathcal{S}}P_{\mathrm{S}}$ zusammengefasst.

2.3.5 Laufzeitoptimierung

Damit die Laufzeit der Simulation möglichst kurz ist, muss die Berechnung der Messpunkte des 2D-Lasertriangulationssensors optimiert werden. Wie in Abbildung 2.4 ersichtlich ist, vermisst der 2D-Lasertriangulationssensor zu einem Zeitpunkt nur einen kleinen Teil des Objektes. Es werden aber alle möglichen Schnittpunkte mit allen Dreiecken des kompletten Objektes berechnet, siehe Abschnitt 2.3.3.

Das Messfeld des Sensors liegt in der xy-Ebene des Sensorkoordinatensystems $\mathcal{S}$. Diese Tatsache kann verwendet werden, um die Anzahl der zu betrachtenden Dreiecke zu

reduzieren, indem die Eckpunkte $\left({}_{\mathcal{S}}\mathbf{d}^j_{\mathrm{O},1}, {}_{\mathcal{S}}\mathbf{d}^j_{\mathrm{O},2}, {}_{\mathcal{S}}\mathbf{d}^j_{\mathrm{O},3}\right)$ relativ zu dieser Ebene betrachtet werden. Wenn das in das Sensorkoordinatensystem $\mathcal{S}$ transformierte Dreieck diese Ebene nicht schneidet, gilt

$$\operatorname{sgn}\left({}_{\mathcal{S}}\mathbf{d}^j_{\mathrm{O},1}\right)_z = \operatorname{sgn}\left({}_{\mathcal{S}}\mathbf{d}^j_{\mathrm{O},2}\right)_z = \operatorname{sgn}\left({}_{\mathcal{S}}\mathbf{d}^j_{\mathrm{O},3}\right)_z . \tag{2.44}$$

Wenn es im Gegensatz dazu die Ebene schneidet, sind die Vorzeichen der z-Komponenten der Eckpunkte ${}_{\mathcal{S}}\mathbf{d}^j_{\mathrm{O},i}$, $i = 1, 2, 3$ unterschiedlich. Mit dieser Unterscheidung kann das Objekt auf die Dreiecke reduziert werden, welche die xy-Ebene schneiden, ohne dass dies Einfluss auf die Genauigkeit der Modellierung hat.

3 Posenbestimmung

Dieses Kapitel handelt von der Positions- und Orientierungsbestimmung des Werkstückes im Arbeitsraum des Roboters. Die genaue Pose soll dabei aus den Messdaten des 2D-Lasertriangulationssensors bestimmt werden. In dieser Arbeit wird dafür eine Variante des Iterative-Closest-Point-Algorithmus (ICP-Algorithmus) verwendet. In Abschnitt 3.1 wird die Funktionsweise des ICP-Algorithmus zusammengefasst. Um den Algorithmus für die Posenbestimmung verwenden zu können, werden die notwendigen Erweiterungen in Abschnitt 3.2 eingeführt. Danach werden in Abschnitt 3.3 die benötigten Schritte für die Implementierung und Laufzeitoptimierung des Algorithmus beschrieben. Am Schluss werden in Abschnitt 3.4 verschiedene ICP-Algorithmen in einer Simulation verglichen.

3.1 Iterative-Closest-Point-Algorithmus (ICP-Algorithmus)

Das Ziel des ICP-Algorithmus ist es, die bestmögliche Überlappung zwischen zwei Punktwolken im $\mathbb{R}^3$ mithilfe eines iterativen Prozesses zu erreichen, siehe Besl et al. [23]. Deshalb wird dieser auch als Punkt-zu-Punkt-ICP-Algorithmus (PzP-ICP-Algorithmus) bezeichnet. Im Folgenden werden in Abschnitt 3.1.1 zunächst die benötigen Definitionen und Funktionen eingeführt und danach der iterative Prozess in Abschnitt 3.1.2 erläutert.

3.1.1 Definitionen

Punktwolken werden allgemein als Menge von Punkten $\mathbf{p}^i\,,\ i = 1, 2, \ldots, N$ in der Form

$$P := \left\{\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^N\right\} \tag{3.1}$$

dargestellt. Für den ICP-Algorithmus werden Punkte aus unterschiedlichen Punktwolken verglichen. Aus diesem Grund sind die Punkte in der Punktwolke nummeriert um eine fixe Reihenfolge zu gewährleisten, siehe (3.1). Die Translation und Rotation einer Punktwolke P erfolgt mit einer homogenen Transformation $\mathbf{H}$ und der Funktion $f_{\mathrm{H}}(\mathbf{H}, P)$ aus (2.10).

Jener Punkt aus der Punktwolke P_b, welcher den minimalen Abstand zu einem gegebenen Punkt $\mathbf{p}_a$ hat, wird mit der Funktion

$$f_{\mathrm{dis}}(\mathbf{p}_a, P_b) := \underset{\mathbf{p}_b \in P_b}{\arg\min} \|\mathbf{p}_a - \mathbf{p}_b\| \tag{3.2}$$

bestimmt. Die Funktion $f_{\mathrm{Dis}}(P_a, P_b)$ sucht mithilfe der Funktion aus (3.2) für alle Punkte $\mathbf{p} \in P_a$ jenen Punkt aus P_b mit dem geringsten Abstand, d. h. es gilt

$$f_{\mathrm{Dis}}(P_a, P_b) := \{f_{\mathrm{dis}}(\mathbf{p}_a, P_b) \mid \forall \mathbf{p}_a \in P_a\}\ . \tag{3.3}$$

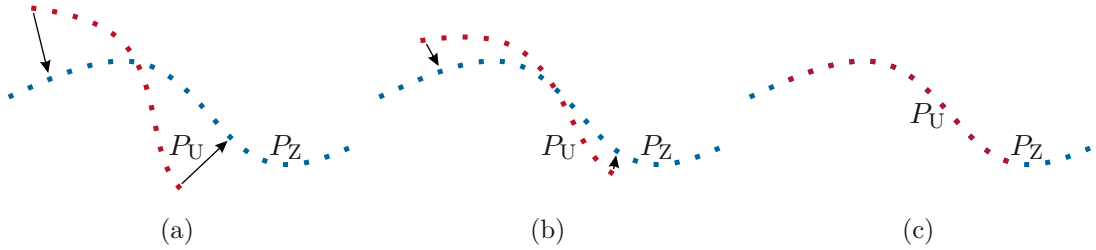


Abbildung 3.1: Schematische Darstellung der Funktionsweise des ICP-Algorithmus mit der Zielpunktwolke P_Z und der Ursprungspunktwolke P_U, (a) erste Iteration, (b) Zwischenschritt, (c) Ergebnis.

Als Metrik des Abstandes zwischen zwei Punktwolken P_a und P_b, welche die gleiche Anzahl an Punkten N haben, wird die mittlere quadratische Abweichung

$$f_E(P_a, P_b) := \frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{p}_a^i - \mathbf{p}_b^i \right\| \tag{3.4}$$

verwendet. Für eine bestmögliche Überlappung zweier Punktwolken P_a, P_b wird die Punktwolke P_a durch die homogene Transformation $\mathbf{H}$ derart transformiert, sodass die Metrik des Abstandes (3.4) minimiert wird. Dabei müssen die Punktwolken die gleiche Anzahl an Punkten N besitzen. Das Optimierungsproblem ergibt sich in der Form

$$\mathbf{H} = \underset{\mathbf{H} \in SE(3)}{\arg\min}\, f_O(P_a, P_b, \mathbf{H}) \tag{3.5a}$$

$$\text{mit} \quad f_O(P_a, P_b, \mathbf{H}) := f_E(f_H(\mathbf{H}, P_a), P_b)\,, \tag{3.5b}$$

in Abhängigkeit der homogenen Transformation $\mathbf{H}$. Für eine geschlossene Lösung dieses Optimierungsproblems wird auf die Arbeit von Horn et al. [24] verwiesen.

3.1.2 Funktionsweise

Das Ziel des ICP-Algorithmus ist es, zwei Punktwolken mit im Allgemeinen unterschiedlicher Anzahl an Punkten bestmöglich in Übereinstimmung zu bringen. Dabei wird eine Punktwolke (Zielpunktwolke P_Z) festgehalten und die andere Punktwolke (Ursprungspunktwolke P_U) wird mithilfe eines iterativen Ablaufs solange transformiert bis es zu einer vorgegebenen Übereinstimmung kommt. In Abbildung 3.1 ist der Ablauf des Algorithmus schematisch dargestellt. Jede Iteration kann in vier Schritte aufgeteilt werden, welche wiederholt werden bis eine vorgegebene Toleranz erreicht ist. Die Iterationen werden mit k gekennzeichnet und für die Initialisierung $k = 0$ wird die Punktwolke P_k, welche in jeder Iteration transformiert wird, mit der Ursprungspunktwolke initialisiert, d. h. $P_0 = P_U$. Die Iteration beginnt mit $k = 1$.

1. Berechnen der nächstgelegenen Punkte der Zielpunktwolke P_Z für alle Punkte aus P_{k-1}, d. h. $P_{Z,k} = f_{Dis}(P_{k-1}, P_Z)$ unter Verwendung von (3.3).
2. Lösen des Optimierungsproblems $\mathbf{H}_0^k = \min_{\mathbf{H} \in SE(3)} f_O(P_0, P_{Z,k}, \mathbf{H})$ aus (3.5).

3. Anwenden der optimalen Transformation mithilfe von $P_k = f_{\mathrm{H}}\left(\mathbf{H}_0^k, P_0\right)$, siehe (2.10), und berechnen des Fehlers $d_k = f_{\mathrm{E}}(P_k, P_{\mathrm{Z},k})$ gemäß (3.4).

4. Wenn die Änderung des Fehlers eine vorgegebene Toleranz τ unterschreitet, d. h. $|d_k - d_{k-1}| < \tau$, so terminiert der Algorithmus. Andernfalls wird k inkrementiert und der Ablauf bei Schritt 1 fortgesetzt.

Das Ergebnis des Algorithmus nach der k-ten Iteration ist die Transformation $\mathbf{H}_0^k$, welche die Ursprungspunktwolke P_{U} transformiert, um eine bestmögliche Überlappung mit der Zielpunktwolke P_{Z} zu erhalten.

3.2 Adaption für die Posenbestimmung des Objektes

In diesem Abschnitt wird der ICP-Algorithmus für die Posenbestimmung des Werkstückes erweitert. Dafür wird zuerst in Abschnitt 3.2.1 das Konzept der Posenbestimmung erklärt. In Abschnitt 3.2.2 wird der Approximierte-Punkt-zu-Ebenen-ICP-Algorithmus (ApP-zE-ICP-Algorithmus) eingeführt, danach wird in Abschnitt 3.2.3 die Bestimmung der nächstgelegenen Punkte beschrieben. Die zugehörigen Abbruchbedingungen werden in Abschnitt 3.2.4 erläutert. Schlussendlich wird in Abschnitt 3.2.5 der Ablauf der Posenbestimmung erklärt.

3.2.1 Konzept der Posenbestimmung

Aufgrund des Modellfehlers des Roboters, siehe Abschnitt 2.2.4, stimmt die Pose des Industrieroboters mit den nominellen Parametern nicht mit der Pose des realen Industrieroboters zusammen. In dieser Arbeit werden die realen Koordinatensysteme von jenen Koordinatensystemen, welche sich durch nominelle Parameter ergeben, durch eine Tilde $(\tilde{\ })$ gekennzeichnet.

In Abbildung 3.2 ist die Pose des nominellen Endeffektorkoordinatensystems $\mathcal{E}$ sowie die Pose des realen Endeffektorkoordinatensystems $\tilde{\mathcal{E}}$ dargestellt. Dabei ist zu beachten, dass nur die reale Pose in der Simulation existiert und dem Regler nicht bekannt ist. Die nominelle Pose ergibt sich aus den nominellen Parametern und ist dem Regler bekannt. Das Koordinatensystem $\mathcal{O}$ bezeichnet die nominelle Pose des Objektes am Endeffektor des Roboters mit nominellen Parametern und das Koordinatensystem $\tilde{\mathcal{O}}$ die reale Pose des Objektes am Endeffektor. Der Ursprung des Objekts wird so gewählt, dass es sich mit dem Endeffektorkoordinatensystem überlappt wenn es am Endeffektor montiert ist siehe Abbildung 3.2. Dies gilt für das reale Modell ${}^{\tilde{\mathcal{O}}}_{\tilde{\mathcal{E}}}\mathbf{H} = \mathbf{E}$, sowie für das nominelle Modell ${}^{\mathcal{O}}_{\mathcal{E}}\mathbf{H} = \mathbf{E}$. Das Symbol $\mathbf{E}$ bezeichnet dabei eine Einheitsmatrix entsprechender Dimension. Das Ziel dieser Arbeit ist es, dass die reale Pose des Objektes $\tilde{\mathcal{O}}$ absolut genau einer Trajektorie folgt. Voraussetzung dafür ist es, die reale Pose des Objektes $\tilde{\mathcal{O}}$ exakt zu bestimmen.

In Abbildung 3.3 ist das Konzept zur Posenbestimmung illustriert. Der 2D-Lasertriangulationssensor vermisst das Objekt an der realen Pose und liefert die Messpunkte ${}_{\mathcal{S}}P$. Durch die bekannte Pose des Sensors und der nominellen direkten Kinematik des Roboters

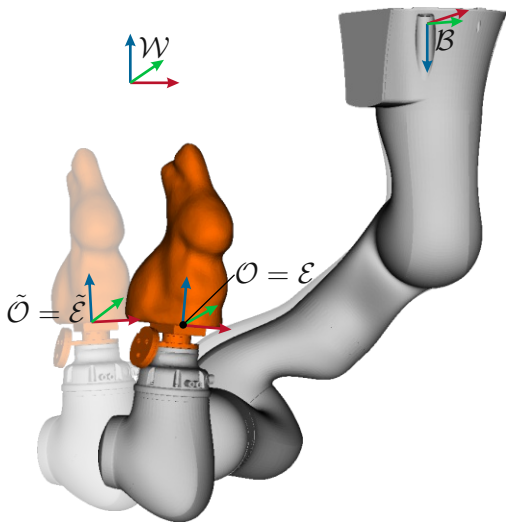


Abbildung 3.2: Darstellung der nominellen Roboterpose $\mathcal{E}, \mathcal{O}$ und realen Roboterpose $\tilde{\mathcal{E}}, \tilde{\mathcal{O}}$.

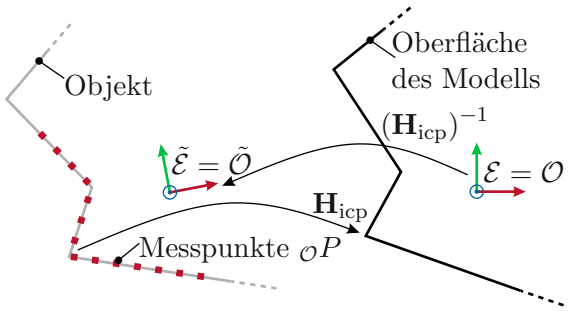


Abbildung 3.3: Konzept der Posenbestimmung mit dem Objekt an der realen Pose $\tilde{\mathcal{O}}$ und der diskreten Oberfläche des Modells des Objektes an der nominellen Pose $\mathcal{O}$.

können die Messpunkte auf die bekannte nominelle Pose des Objektkoordinatensystem $\mathcal{O}$ bezogen werden ${}_{\mathcal{O}}P$.

Unter der Annahme, dass aus den Messdaten mithilfe eines Algorithmus die Transformation $\mathbf{H}_{\text{icp}}$ von den Messpunkten ${}_{\mathcal{O}}P$ zur diskreten Oberfläche des Modells des Objektes berechnet werden kann, ergibt sich die reale Pose des Objektes relativ zur Basis des Industrieroboters in der Form

$$ {}^{\tilde{\mathcal{O}}}_{\mathcal{B}}\mathbf{H} = {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{H}\, {}^{\mathcal{O}}_{\mathcal{E}}\mathbf{H}\, (\mathbf{H}_{\text{icp}})^{-1} \ . \tag{3.6} $$

3.2.2 Lineare Approximation des Punkt-zu-Ebenen-ICP-Algorithmus

Beim originalen ICP-Algorithmus wird die Metrik des Abstandes (3.4) zwischen den Punktwolken berechnet, weshalb dieser als Punkt-zu-Punkt-ICP-Algorithmus (PzP-ICP-Algorith-

mus) bezeichnet wird. Wenn die Metrik zwischen Punkten und einer Ebene berechnet wird, wird dieser als Punkt-zu-Ebenen-ICP-Algorithmus (PzE-ICP-Algorithmus) bezeichnet.

Wie in Abschnitt 2.1.3 beschrieben, ist zu jedem Dreieck der diskreten Oberfläche eines Objektes ein zugehöriger Normalvektor definiert. Um diese zusätzliche Information im Algorithmus zu verwenden, wird in diesem Abschnitt eine Variante des Punkt-zu-Ebenen-ICP-Algorithmus (ICP-Algorithmus) hergeleitet, siehe Low [25]. Die Arbeit [25] beschreibt den Approximierte-Punkt-zu-Ebenen-ICP-Algorithmus (ApPzE-ICP-Algorithmus), welcher durch eine Approximation der Fehlermetrik eine schnellere und genauere Konvergenz als der PzP-ICP-Algorithmus hat. Die Optimierung der Fehlermetrik entspricht dem zweiten Schritt jeder Iteration des PzP-ICP-Algorithmus, vgl. Abschnitt 3.1.2. Im Folgenden werden die Ergebnisse aus [25] zusammengefasst.

Für den Algorithmus muss zu jedem Punkt aus der Zielpunktwolke P_Z ein dazugehöriger Normalvektor existieren N_Z. In diesem Abschnitt wird angenommen, dass die zur Punktwolke P_{k-1} nächstgelegenen Punkte der Zielpunktwolke $P_{\mathrm{Z},k} := \left\{\mathbf{p}^1_{\mathrm{Z},k}, \mathbf{p}^2_{\mathrm{Z},k}, \ldots, \mathbf{p}^N_{\mathrm{Z},k}\right\}$ und Normalvektoren $N_{\mathrm{Z},k} := \left\{\mathbf{n}^1_{\mathrm{Z},k}, \mathbf{n}^2_{\mathrm{Z},k}, \ldots, \mathbf{n}^N_{\mathrm{Z},k}\right\}$ schon berechnet wurden. Die Berechnungsvorschrift für diese Punkte und Normalvektoren wird in Abschnitt 3.2.3 beschrieben.

Im Gegensatz zum PzP-ICP-Algorithmus aus Abschnitt 3.1 wird beim PzE-ICP-Algorithmus nicht der Abstand zwischen den Punkten minimiert, sondern der Normalabstand zwischen den Punkten und jener Ebene, welche von dem Punkt und dem dazugehörigen Normalvektor aufgespannt wird. Dadurch ändert sich das Optimierungsproblem, welches abhängig von $\mathbf{H}^k_{k-1}$ minimiert wird, von (3.5) zu

$$\mathbf{H}^k_{k-1} = \underset{\mathbf{H}\in\mathrm{SE}(3)}{\arg\min}\, f_\mathrm{O}(P_{k-1}, P_{\mathrm{Z},k}, N_{\mathrm{Z},k}, \mathbf{H}) \tag{3.7a}$$

$$\text{mit} \quad f_\mathrm{O}(P_{k-1}, P_{\mathrm{Z},k}, N_{\mathrm{Z},k}, \mathbf{H}) := \sum_{i=1}^{N} \left\| \left(f_\mathrm{H}\left(\mathbf{H}, \mathbf{p}^i_{k-1}\right) - \mathbf{p}^i_{\mathrm{Z},k} \right) \cdot \mathbf{n}^i_{\mathrm{Z},k} \right\| . \tag{3.7b}$$

Es ist dabei zu beachten, dass bei (3.5) die Fehlermetrik zwischen P_0 und $P_{\mathrm{Z},k}$ minimiert wird aber in (3.7) zwischen P_{k-1} und $P_{\mathrm{Z},k}$. Aus diesem Grund muss für das Ergebnis die Transformation $\mathbf{H}^k_0$ gemäß

$$\mathbf{H}^k_0 = \mathbf{H}^{k-1}_0 \, \mathbf{H}^k_{k-1} \tag{3.8}$$

bei jeder Iteration zusätzlich berechnet werden. Die homogene Transformation

$$\mathbf{H}^k_{k-1} = \mathbf{H}_{\mathrm{Tz},d_\mathrm{z}} \, \mathbf{H}_{\mathrm{Ty},d_\mathrm{y}} \, \mathbf{H}_{\mathrm{Tx},d_\mathrm{x}} \, \mathbf{H}_{\mathrm{Rz},\alpha_\mathrm{z}} \, \mathbf{H}_{\mathrm{Ry},\alpha_\mathrm{y}} \, \mathbf{H}_{\mathrm{Rx},\alpha_\mathrm{x}} \tag{3.9}$$

wird durch Aneinanderreihen der entsprechenden Transformationen der Variablen d_i und α_i , $i = x, y, z$ gebildet.

Bei dem Optimierungsproblem (3.7) handelt es sich um eine Optimierung nach dem kleinsten Quadrat der Variablen d_i und α_i. Diese Optimierungsvariablen sind Argumente von trigonometrischen Funktionen, wodurch es sich um eine nichtlineare Optimierung handelt und nicht effizient gelöst werden kann. Mit der Annahme, dass die Transformation $\mathbf{H}^k_{k-1}$ nur kleine Winkel aufweist, d. h. $\alpha_i \approx 0$, kann (3.9) durch eine lineare Approximation

in Abhängigkeit des Zustandes $\mathbf{x} = \begin{bmatrix}\alpha_\mathrm{x} & \alpha_\mathrm{y} & \alpha_\mathrm{z} & d_\mathrm{x} & d_\mathrm{y} & d_\mathrm{z}\end{bmatrix}^\mathrm{T}$ als Matrix

$$\hat{\mathbf{M}}_{k-1}^{k}(\mathbf{x}) = \begin{bmatrix} 1 & -\alpha_\mathrm{z} & \alpha_\mathrm{y} & d_\mathrm{x} \\ \alpha_\mathrm{z} & 1 & -\alpha_\mathrm{x} & d_\mathrm{y} \\ -\alpha_\mathrm{y} & \alpha_\mathrm{x} & 1 & d_\mathrm{z} \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.10}$$

angeschrieben werden. In (3.10) und im Folgenden kennzeichnet das Symbol ($\hat{}$) Approximationen oder Schätzungen. Ein Vergleich des Rotationsteils der Transformation (3.10) mit der schiefsymmetrischen Matrix (2.19) zeigt, dass

$$f_\mathrm{R}\left(\hat{\mathbf{M}}_{k-1}^{k}(\mathbf{x})\right)\mathbf{p} = \left(\mathbf{E} + f_\mathrm{S}\left(\begin{bmatrix}\alpha_\mathrm{x} & \alpha_\mathrm{y} & \alpha_\mathrm{z}\end{bmatrix}^\mathrm{T}\right)\right)\mathbf{p} = \mathbf{p} + \begin{bmatrix}\alpha_\mathrm{x} & \alpha_\mathrm{y} & \alpha_\mathrm{z}\end{bmatrix}^\mathrm{T} \times \mathbf{p} \tag{3.11}$$

gilt. Durch Einsetzen von (3.10) in (3.7) kann die Transformation ausmultipliziert werden und ergibt das approximierte lineare Optimierungsproblem

$$\hat{\mathbf{M}}_{k-1}^{k}(\mathbf{x}) = \underset{\mathbf{x}\in\mathbb{R}^6}{\arg\min}\, f_\mathrm{O}\left(P_{k-1}, P_{\mathrm{Z},k}, N_{\mathrm{Z},k}, \hat{\mathbf{M}}(\mathbf{x})\right)$$

$$\text{mit} \quad f_\mathrm{O}\left(P_{k-1}, P_{\mathrm{Z},k}, N_{\mathrm{Z},k}, \hat{\mathbf{M}}(\mathbf{x})\right) := \sum_{i=1}^{N}\left\|\left(f_\mathrm{H}\left(\hat{\mathbf{M}}(\mathbf{x}), \mathbf{p}_{k-1}^{i}\right) - \mathbf{p}_{\mathrm{Z},k}^{i}\right)\cdot\mathbf{n}_{\mathrm{Z},k}^{i}\right\| =$$

$$\sum_{i=1}^{N}\left\|\left(f_\mathrm{R}\left(\hat{\mathbf{M}}(\mathbf{x})\right)\mathbf{p}_{k-1}^{i} + f_\mathrm{d}\left(\hat{\mathbf{M}}(\mathbf{x})\right) - \mathbf{p}_{\mathrm{Z},k}^{i}\right)\cdot\mathbf{n}_{\mathrm{Z},k}^{i}\right\| =$$

$$\sum_{i=1}^{N}\left\|\mathbf{p}_{k-1}^{i}\cdot\mathbf{n}_{\mathrm{Z},k}^{i} + \left(\begin{bmatrix}\alpha_\mathrm{x} & \alpha_\mathrm{y} & \alpha_\mathrm{z}\end{bmatrix}^\mathrm{T}\times\mathbf{p}_{k-1}^{i}\right)\cdot\mathbf{n}_{\mathrm{Z},k}^{i} + \begin{bmatrix}d_\mathrm{x} & d_\mathrm{y} & d_\mathrm{z}\end{bmatrix}^\mathrm{T}\cdot\mathbf{n}_{\mathrm{Z},k}^{i} - \mathbf{p}_{\mathrm{Z},k}^{i}\cdot\mathbf{n}_{\mathrm{Z},k}^{i}\right\|. \tag{3.12}$$

Mit der Eigenschaft der zyklischen Vertauschung beim Spatprodukt

$$(\mathbf{a}\times\mathbf{b})\cdot\mathbf{c} = (\mathbf{c}\times\mathbf{a})\cdot\mathbf{b} = (\mathbf{b}\times\mathbf{c})\cdot\mathbf{a}\,, \tag{3.13}$$

mit $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^3$ kann (3.12) in Abhängigkeit von $\mathbf{x} = \begin{bmatrix}\alpha_\mathrm{x} & \alpha_\mathrm{y} & \alpha_\mathrm{z} & d_\mathrm{x} & d_\mathrm{y} & d_\mathrm{z}\end{bmatrix}^\mathrm{T}$ in der Form

$$\mathbf{x}_\mathrm{A} = \underset{\mathbf{x}\in\mathbb{R}^6}{\arg\min}\,\|\mathbf{A}_\mathrm{A}\mathbf{x} - \mathbf{b}_\mathrm{A}\| \tag{3.14a}$$

$$\text{mit} \quad \mathbf{A}_\mathrm{A} = \begin{bmatrix} \left(\mathbf{p}_{k-1}^{1}\times\mathbf{n}_{\mathrm{Z},k}^{1}\right)^\mathrm{T} & \left(\mathbf{n}_{\mathrm{Z},k}^{1}\right)^\mathrm{T} \\ \left(\mathbf{p}_{k-1}^{2}\times\mathbf{n}_{\mathrm{Z},k}^{2}\right)^\mathrm{T} & \left(\mathbf{n}_{\mathrm{Z},k}^{2}\right)^\mathrm{T} \\ \vdots & \vdots \\ \left(\mathbf{p}_{k-1}^{N}\times\mathbf{n}_{\mathrm{Z},k}^{N}\right)^\mathrm{T} & \left(\mathbf{n}_{\mathrm{Z},k}^{N}\right)^\mathrm{T} \end{bmatrix} \tag{3.14b}$$

$$\mathbf{b}_\mathrm{A} = \begin{bmatrix} \left(\mathbf{p}_{\mathrm{Z},k}^{1} - \mathbf{p}_{k-1}^{1}\right)\cdot\mathbf{n}_{\mathrm{Z},k}^{1} \\ \left(\mathbf{p}_{\mathrm{Z},k}^{2} - \mathbf{p}_{k-1}^{2}\right)\cdot\mathbf{n}_{\mathrm{Z},k}^{2} \\ \vdots \\ \left(\mathbf{p}_{\mathrm{Z},k}^{N} - \mathbf{p}_{k-1}^{N}\right)\cdot\mathbf{n}_{\mathrm{Z},k}^{N} \end{bmatrix} \tag{3.14c}$$

zusammengefasst werden. Mithilfe der linken Pseudoinversen von $\mathbf{A}_\mathrm{A}$, definiert als

$$\mathbf{A}_\mathrm{A}^{l\dagger} = \left((\mathbf{A}_\mathrm{A})^\mathrm{T}\mathbf{A}_\mathrm{A}\right)^{-1}(\mathbf{A}_\mathrm{A})^\mathrm{T}, \tag{3.15}$$

wird die Lösung $\mathbf{x}_\mathrm{A}$ gemäß

$$\mathbf{x}_\mathrm{A} = \mathbf{A}_\mathrm{A}^{l\dagger}\,\mathbf{b}_\mathrm{A} \tag{3.16}$$

berechnet. Um eine gültige homogene Transformation zu erhalten, wird schlussendlich die Lösung $\mathbf{x}_\mathrm{A}$ von (3.16) in die ursprüngliche homogene Transformation $\mathbf{H}_{k-1}^{k}$ gemäß (3.9) eingesetzt.

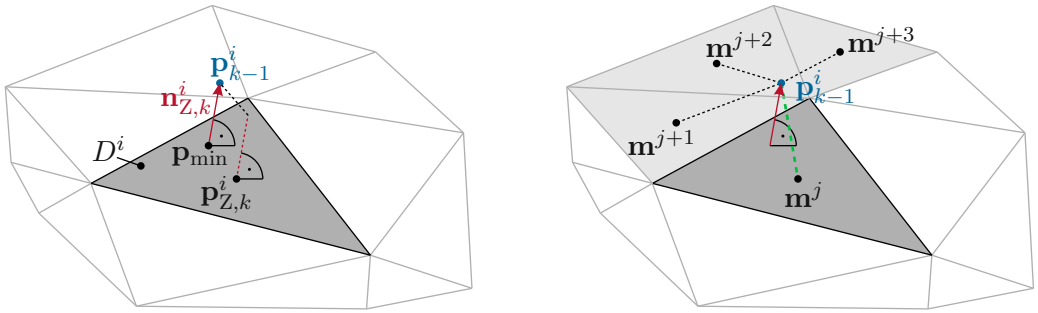
3.2.3 Approximation der Bestimmung der nächstgelegenen Punkte

Dieser Abschnitt beschreibt die Bestimmung der zur iterierenden Punktwolke $P_{k-1} = \left\{\mathbf{p}_{k-1}^1, \mathbf{p}_{k-1}^2, \ldots, \mathbf{p}_{k-1}^N\right\}$ gehörenden nächstgelegenen Punkte $P_{\mathrm{Z},k} = \left\{\mathbf{p}_{\mathrm{Z},k}^1, \mathbf{p}_{\mathrm{Z},k}^2, \ldots, \mathbf{p}_{\mathrm{Z},k}^N\right\}$. Dies entspricht dem ersten Schritt jeder Iteration des PzP-ICP-Algorithmus, vgl. Abschnitt 3.1.2. Beim PzP-ICP-Algorithmus werden die am nächsten liegenden Punkte $P_{\mathrm{Z},k} = \left\{\mathbf{p}_{\mathrm{Z},k}^1, \mathbf{p}_{\mathrm{Z},k}^2, \ldots, \mathbf{p}_{\mathrm{Z},k}^N\right\}$ direkt aus den Punkten der Zielpunktwolke P_Z gewählt. Für die Berechnung des zum Punkt $\mathbf{p}_{\mathrm{Z},k}^i$ nächstgelegenen Punktes $\mathbf{p}_\mathrm{min}$ wird die normierte Distanz zu jedem Punkt aus der Zielpunktwolke P_Z berechnet. Jener Punkt mit dem minimalen Abstand ist der am nächsten liegende Punkt $\mathbf{p}_\mathrm{min}$.

In dieser Arbeit liegt die Oberfläche des Modells als diskrete Freiformoberfläche vor, welche aus Dreiecken $D = \left\{D^1, D^2, \ldots, D^{N_\mathrm{D}}\right\}$ aufgebaut ist. Für jedes Dreieck D^j, $j = 1, 2, \ldots, N_\mathrm{D}$, ist ein korrespondierender Normalvektor $\mathbf{n}^j$ definiert, welcher normal zum Dreieck steht. In der Arbeit von Li et al. [26] wird beschrieben, wie der am nächsten liegende Punkt exakt in Bezug auf die Oberfläche berechnet werden kann. Dies ist gegenüber der Berechnung vom PzP-ICP-Algorithmus rechenintensiver. Zur Darstellung dieses Sachverhalts werden die benötigten Rechenschritte am Beispiel eines Punktes $\mathbf{p}_{\mathrm{Z},k}^i \in P_{\mathrm{Z},k}$ auf der Freiformoberfläche im Folgenden grob beschrieben.

Zunächst ist nicht bekannt, auf welchem Dreieck D^j, $j = 1, 2, \ldots, N_\mathrm{D}$, der Freiformoberfläche sich der nächstgelegene Punkt $\mathbf{p}_\mathrm{min}$ befindet. Aus diesem Grund muss für jedes Dreieck D^j der zugehörige Punkt mit dem Minimalabstand $\mathbf{p}_\mathrm{min}^j$ berechnet werden. Diese Punkte werden zu einer Punktwolke $P_\mathrm{min} = \left\{\mathbf{p}_\mathrm{min}^1, \mathbf{p}_\mathrm{min}^2, \ldots, \mathbf{p}_\mathrm{min}^{N_\mathrm{D}}\right\}$ zusammengefasst. Danach erfolgt die Berechnung des tatsächlichen Punktes mit Minimalabstand $\mathbf{p}_\mathrm{min}$ analog zum PzP-ICP-Algorithmus $\mathbf{p}_\mathrm{min} = f_\mathrm{dis}\left(\mathbf{p}_{\mathrm{Z},k}^i, P_\mathrm{min}\right)$, siehe (3.2). Die Punktwolke P_min enthält für verschiedene Punkte $\mathbf{p}_{\mathrm{Z},k}^i$ im Allgemeinen immer andere Punkte mit dem Minimalabstand $P_\mathrm{min} = \left\{\mathbf{p}_\mathrm{min}^1, \mathbf{p}_\mathrm{min}^2, \ldots, \mathbf{p}_\mathrm{min}^{N_\mathrm{D}}\right\}$. Deshalb muss die Berechnung für jeden Punkt $\mathbf{p}_{\mathrm{Z},k}^i \in P_{\mathrm{Z},k}$ bei jeder Iteration k des ICP-Algorithmus wiederholt werden. Dadurch sind weitere Optimierungen des Algorithmus schwierig.

In der Arbeit [26] werden Objekte verwendet, welche aus einer relativ kleinen Anzahl an Dreiecken (zwischen 12 und 1825) bestehen. Dadurch ist der Rechenaufwand begrenzt. Das in der vorliegenden Arbeit verwendete Objekt besteht jedoch aus $1 \cdot 10^5$ Dreiecken, wodurch eine exakte Berechnung des Punktes $\mathbf{p}_\mathrm{min}$ mit der Methode [26] langsam und



(a) (b)

Abbildung 3.4: Schematische Darstellung der Approximation der Bestimmung der nächstgelegenen Punkte, (a) konstanter Normalabstand am Dreieck D^i, (b) Approximation durch die Mittelpunkte $\mathbf{m}^j$, $j = 1, 2, \ldots, N_{\mathrm{D}}$.

ineffizient ist. Aus diesem Grund ist dieser Algorithmus nicht für die Regelung eines Industrieroboters geeignet.

Bei dem ApPzE-ICP-Algorithmus wird der Normalabstand zwischen der iterierenden Punktwolke $P_{k-1} = \left\{\mathbf{p}^1_{k-1}, \mathbf{p}^2_{k-1}, \ldots, \mathbf{p}^N_{k-1}\right\}$ und den nächstgelegenen Ebenen minimiert, vgl. (3.14). Diese Ebenen werden von der Punktwolke $P_{\mathrm{Z},k} = \left\{\mathbf{p}^1_{\mathrm{Z},k}, \mathbf{p}^2_{\mathrm{Z},k}, \ldots, \mathbf{p}^N_{\mathrm{Z},k}\right\}$ und den korrespondierenden Normalvektoren aufgespannt. Solange sich der Punkt $\mathbf{p}^i_{\mathrm{Z},k}$ auf der Ebene aufgespannt von $\mathbf{n}^i_{\mathrm{Z},k}$ befindet ist der Normalabstand $\left(\mathbf{p}^i_{k-1} - \mathbf{p}^i_{\mathrm{Z},k}\right) \cdot \mathbf{n}^i_{\mathrm{Z},k}$ gleich wie der Normalabstand zum nächstgelegenen Punkt $\left(\mathbf{p}^i_{k-1} - \mathbf{p}_{\min}\right) \cdot \mathbf{n}^i_{\mathrm{Z},k}$, siehe Abbildung 3.4a. Folglich ist das Ergebnis des Optimierungsproblems dasselbe wie beim ApPzE-ICP-Algorithmus, solange das nächstgelegene Dreieck D^i mit dem korrekten Normalvektor $\mathbf{n}^i_{\mathrm{Z},k}$ gefunden wird. Für das Optimierungsproblem (3.14) kann ein beliebiger Punkt auf der Ebene, in der das Dreiecks liegt, verwendet werden. In der vorliegenden Arbeit wird angenommen, dass das nächstgelegene Dreieck durch eine Approximation gefunden werden kann. Dafür wird jedes Dreieck $D = \left\{D^1, D^2, \ldots, D^{N_{\mathrm{D}}}\right\}$ durch den korrespondierenden Mittelpunkt des Dreiecks $M = \left\{\mathbf{m}^1, \mathbf{m}^2, \ldots, \mathbf{m}^{N_{\mathrm{D}}}\right\}$ repräsentiert. Durch diese Approximation kann das nächstgelegene Dreieck, analog zum PzP-ICP-Algorithmus, durch die Funktion $M_{\mathrm{Z},k} = f_{\mathrm{Dis}}(P_{k-1}, M)$ aus (3.3) berechnet werden. Die Approximation ist in Abbildung 3.4b anhand einiger Dreiecke vereinfacht dargestellt. Die so berechneten Mittelpunkte $M_{\mathrm{Z},k}$ entsprechen den nächstgelegenen Punkten $P_{\mathrm{Z},k} = M_{\mathrm{Z},k}$. Die Menge der Normalvektoren $N_{\mathrm{Z},k}$ sind die zu den Mittelpunkten $M_{\mathrm{Z},k}$ gehörenden Normalvektoren.

Die Genauigkeit der Approximation ist von der Geometrie, dem Aufbau und der Anzahl an Dreiecken des Objektes abhängig. Wenn die Dichte der Dreiecke verhältnismäßig groß ist, funktioniert diese Approximation sehr gut. Der Rechenaufwand zur Bestimmung der korrespondierenden Punkten wird dadurch wieder auf das Niveau des PzP-ICP-Algorithmus reduziert, vgl. (3.3). Jedoch erlaubt der zusätzliche Normalvektor die Verwendung des ApPzE-ICP-Algorithmus.

3.2.4 Abbruchbedingungen

Damit der iterative Prozess der Posenbestimmung beendet werden kann, sind Abbruchbedingungen notwendig. Diese Bedingungen werden im vierten Schritt in jeder Iteration des PzP-ICP-Algorithmus geprüft, vgl. Abschnitt 3.1.2. Beim ICP-Algorithmus wird dieser abgebrochen, wenn die Änderung der Fehlermetrik (3.4) von einer Iteration zur nächsten Iteration unter einer vorgegebenen Toleranz fällt, siehe Abschnitt 3.1.2. Um die Fehleranfälligkeit zu verringern und die Zeitdauer des Algorithmus zu begrenzen, werden in diesem Abschnitt zusätzliche Bedingungen eingeführt.

Die Fehlermetrik der Optimierung für den ApPzE-ICP-Algorithmus (3.14) verwendet den Normalabstand und dadurch ist es nicht möglich aus dieser Metrik den Versatz, d. h. die Translation und die Rotation der Punktwolke abzuschätzen. Auch sollen durch die Abbruchbedingungen unplausible Ergebnisse erkannt werden, wenn ein vorgegebener Schwellenwert bei der Translation oder Rotation überschritten wird.

Aus diesem Grund wird eine zusätzliche Fehlermetrik für die Abbruchbedingungen eingeführt. Diese beurteilt direkt die Translation und Rotation zwischen zwei Punktwolken. Der translatorische Fehler

$$f_{\mathrm{E,T}}\left(\mathbf{H}_a^b, P_a\right) := \left\| \frac{1}{N_a} \sum_{i=1}^{N_a} \mathbf{p}_a^i - \mathbf{p}_b^i \right\|, \tag{3.17}$$

mit $\mathbf{p}_a^i \in P_a$ und $\mathbf{p}_b^i \in f_{\mathrm{H}}\left(\mathbf{H}_a^b, P_a\right)$ entspricht der euklidischen Norm der Differenz des Mittelpunktes der Punktwolken. Der rotatorische Fehler

$$f_{\mathrm{E,R}}\left(\mathbf{H}_a^b\right) := \operatorname{acos}\left(\frac{\operatorname{Tr}\left(f_{\mathrm{R}}\left(\mathbf{H}_a^b\right)\right) - 1}{2}\right) \tag{3.18}$$

entspricht dem Betrag des Winkels der Rotation aus der Rotationsmatrix $\mathbf{H}_a^b$ und wird mithilfe der Spur $\operatorname{Tr}(\cdot)$ der Matrix berechnet, siehe [27]. Die Arkuscosinusfunktion $\operatorname{acos}(\cdot)$ ergibt immer positive Werte, weshalb auch der Winkel der Rotation immer positiv ist.

In dieser Arbeit werden drei Abbruchbedingungen verwendet, die im Folgenden beschrieben werden.

1. **Die iterative Transformation $\mathbf{H}_{k-1}^k$ (3.9) unterschreitet eine festgelegte Toleranz.**

Die Punktwolke für die nächste Iteration berechnet sich zu $P_k = f_{\mathrm{H}}\left(\mathbf{H}_{k-1}^k, P_{k-1}\right)$. Wenn die iterative Transformation $\mathbf{H}_{k-1}^k$ die vorgegebene Toleranz sowohl bei der Translation $\tau_{\mathrm{T,min}}$ und der Rotation $\tau_{\mathrm{R,min}}$ unterschreitet, d. h. es gilt

$$f_{\mathrm{E,T}}\left(\mathbf{H}_{k-1}^k, P_{k-1}\right) < \tau_{\mathrm{T,min}} \ \wedge \ f_{\mathrm{E,R}}\left(\mathbf{H}_{k-1}^k\right) < \tau_{\mathrm{R,min}}\,, \tag{3.19}$$

dann ist das Ergebnis des Algorithmus die homogene Transformation $\mathbf{H}_{\mathrm{icp}} = \mathbf{H}_0^k$.

2. **Die Iteration k erreicht die maximale Iteration $k_{\max}$.**

$$k \geq k_{\max} \tag{3.20}$$

Durch die Begrenzung der Iteration auf eine maximale Iteration $k_{\max}$ wird verhindert, dass der Algorithmus über eine unbestimmt lange Zeit läuft. Dadurch wird die Echtzeitfähigkeit des Algorithmus für den Einsatz in einem geschlossenen Regelkreis ermöglicht. Falls die letzte Iteration erreicht wird, ist das Ergebnis des Algorithmus die homogene Transformation $\mathbf{H}_{\text{icp}} = \mathbf{H}_0^{k_{\max}}$.

3. Die Gesamttransformation $\mathbf{H}_0^k$ (3.8) überschreitet einen maximalen Fehler.

Es wird angenommen, dass der Industrieroboter eine gewisse maximale Ungenauigkeit hat. Um unplausible bzw. ungültige Ergebnisse zu eliminieren, wird eine maximale Translation $\tau_{\text{T,max}}$ und Rotation $\tau_{\text{R,max}}$ eingeführt. Wird einer der Grenzwerte überschritten, d. h. es gilt

$$f_{\text{E,T}}\left(\mathbf{H}_0^k, P_0\right) > \tau_{\text{T,max}} \ \vee \ f_{\text{E,R}}\left(\mathbf{H}_0^k\right) > \tau_{\text{R,max}}\,, \tag{3.21}$$

so wird $\mathbf{H}_0^k$ verworfen und das Ergebnis des Algorithmus ist die homogene Transformation $\mathbf{H}_{\text{icp}} = \mathbf{E}$. Die Einheitsmatrix entspricht einer unveränderten Pose als Ergebnis des Algorithmus.

3.2.5 Ablauf der Posenbestimmung

Die Posenbestimmung erfolgt relativ zur nominellen Pose des Objektes $\mathcal{O}$, da diese durch die nominelle direkte Kinematik ${}^{\mathcal{E}}_{\mathcal{B}}\mathbf{H}(\mathbf{q})$ und die Zusammenhänge ${}^{\mathcal{O}}_{\mathcal{E}}\mathbf{H}$ sowie ${}^{\mathcal{B}}_{\mathcal{W}}\mathbf{H}$ bekannt ist, siehe (2.16).

Die Messdaten des Sensors ${}_{\mathcal{S}}P_{\text{S}}$, siehe Abschnitt 2.3.4, werden über die bekannte Transformation

$$ {}^{\mathcal{S}}_{\mathcal{O}}\mathbf{H}(\mathbf{q}) = \left({}^{\mathcal{O}}_{\mathcal{W}}\mathbf{H}(\mathbf{q})\right)^{-1} {}^{\mathcal{S}}_{\mathcal{W}}\mathbf{H} \tag{3.22a}$$

$$\text{mit} \quad {}^{\mathcal{O}}_{\mathcal{W}}\mathbf{H}(\mathbf{q}) = {}^{\mathcal{B}}_{\mathcal{W}}\mathbf{H}\ {}^{\mathcal{E}}_{\mathcal{B}}\mathbf{H}(\mathbf{q})\ {}^{\mathcal{O}}_{\mathcal{E}}\mathbf{H} \tag{3.22b}$$

auf das Koordinatensystem $\mathcal{O}$ bezogen. Die in das Koordinatensystem $\mathcal{O}$ transformierten Messpunkte ${}_{\mathcal{O}}P_{\text{S}} = f_{\text{H}}\left({}^{\mathcal{S}}_{\mathcal{O}}\mathbf{H}(\mathbf{q}), {}_{\mathcal{S}}P_{\text{S}}\right)$ entsprechen der Ursprungswolke $P_{\text{U}} = {}_{\mathcal{O}}P_{\text{S}}$. Wie in Abschnitt 3.2.3 beschrieben, werden die Mittelpunkte ${}_{\mathcal{O}}M_{\text{O}}$ aller Dreiecke ${}_{\mathcal{O}}D_{\text{O}}$ des Objektes als Zielpunktwolke verwendet, d. h. $P_{\text{Z}} = {}_{\mathcal{O}}M_{\text{O}}$. Die korrespondierenden Normalvektoren der Zielpunktwolke N_{Z} entsprechen den zum Mittelpunkt gehörenden Normalvektoren des Objektes $N_{\text{Z}} = {}_{\mathcal{O}}N_{\text{O}}$.

Für die Initialisierung des iterativen Prozesses wird die Punktwolke P_k mit der Ursprungswolke $P_0 = P_{\text{U}}$ initialisiert. Der Ablauf der Posenbestimmung beginnt mit $k = 1$ und ist wie folgt:

1. Berechnen von $P_{\text{Z},k} = f_{\text{Dis}}(P_{k-1}, P_{\text{Z}})$ gemäß (3.3) und den korrespondierenden Normalvektoren $N_{\text{Z},k}$, siehe Abschnitt 3.2.3.

2. Lösen des Optimierungsproblems $\mathbf{x}_{\text{A}} = \arg\min_{\mathbf{x}\in\mathbb{R}^6} f_{\text{O}}(P_{k-1}, P_{\text{Z},k}, N_{\text{Z},k}, \mathbf{x})$ und berechnen von $\mathbf{H}_{k-1}^k$ und $\mathbf{H}_0^k$, siehe Abschnitt 3.2.2.

3. Berechnen der aktualisierten iterierenden Punktwolke $P_k = f_{\text{H}}\left(\mathbf{H}_{k-1}^k, P_{k-1}\right)$.

4. Prüfen der Abbruchbedingungen (3.19), (3.20) und (3.21). Ist keine der Bedienungen erfüllt, so wird $k = k + 1$ gesetzt und die nächste Iteration mit Schritt 1 gestartet. Andernfalls liegt das Ergebnis des Algorithmus $\mathbf{H}_{\text{icp}}$ vor, siehe Abschnitt 3.2.4

3.3 Implementierung

Um den Algorithmus für die Regelung verwenden zu können, werden in diesem Abschnitt die entsprechenden Details der Implementierung beschrieben. In Abschnitt 3.3.1 wird der k-d-Baum eingeführt, welcher zu einer Verbesserung der Laufzeit führt. Um den Algorithmus mit einer anderen Abtastzeit als die Regelung verwenden zu können, wird in Abschnitt 3.3.2 ein Signalfilter eingeführt. Um den Rechenaufwand des Algorithmus zu verringern, wird in Abschnitt 3.3.3 beschrieben, wie das Ergebnis der vorherigen Iteration verwendet werden kann.

3.3.1 *k*-d-Baum

Um die Performance des Algorithmus weiter zu verbessern, wird für die Bestimmung des am nächsten liegenden Punktes (3.3) die MATLAB-Funktion `knnsearch` verwendet. Diese basiert auf dem k-d-Baum-Algorithmus von Friedman et al. [28]. Dadurch kann die Komplexität gegenüber einer linearen Suche von $\mathcal{O}(N)$ auf $\mathcal{O}(\log(N))$ verringert werden, wobei N der Anzahl an Punkten in der Punktwolke entspricht.

Der k-d-Baum ist ein binärer Baum mit folgendem Aufbau: Der erste Knoten enthält die gesamte Punktwolke. Diese Punktwolke wird durch eine Ebene in zwei gleich große Punktwolken geteilt. Jede Hälfte ist ein neuer Knoten, welcher wieder geteilt wird. Dieser Vorgang wird solange wiederholt bis die Knoten nur noch einen Punkt enthalten. Für die Suche des am nächsten liegenden Punktes müssen nur die Bedingungen der Ebenen kontrolliert werden, wodurch die Suchzeit reduziert wird. Dies funktioniert nur solange die Punktwolke sich nicht verändert. Der k-d-Baum-Algorithmus liefert die selben Ergebnisse wie die lineare Suche, wodurch es zu keiner Beeinträchtigung der Genauigkeit des ICP-Algorithmus kommt.

3.3.2 Filtern des Signales

Das Ergebnis der Posenbestimmung muss aus mehreren Gründen gefiltert werden. Einerseits sind die Ergebnisse der Posenbestimmung des Objektes nicht stetig und deshalb nicht direkt für eine Regelung geeignet. Zusätzlich soll die Änderungsrate der Objektpose mitbetrachtet werden, um eine bessere Regelgüte zu erreichen.

Die verwendeten Abtastzeiten der Posenbestimmung T_{icp} und des Reglers sind T_{R} sind in Tabelle 3.1 zusammengefasst. Aufgrund der unterschiedlichen Abtastzeiten ist ein Filter notwendig damit die Pose für die Regelung verwendet werden kann. Um eine stetige Schätzung mit zugehörigen Ableitungen für die Regelung zu erhalten, wird das Ergebnis der Posenbestimmung $\mathbf{H}_{\text{icp}}$ gefiltert. Eine Interpolation der einzelnen Elemente der homogenen Transformation ist jedoch nicht möglich, da dies zu ungültigen Werten in der Rotationsmatrizen führen würde [29]. Aus diesem Grund wird die homogene Transformation $\mathbf{H}_{\text{icp}}$ auf den translatorischen und den rotatorischen Anteil aufgeteilt.

Tabelle 3.1: Abtastzeiten in der Simulation.

Symbol	Einheit	Wert
T_{icp}	ms	50
T_{R}	µs	125

Tabelle 3.2: Filterkoeffizienten der Filter für die Posenbestimmung.

Symbol	Einheit	Wert
$b_{\text{f},0}$	$1/\text{s}^2$	400
$b_{\text{f},1}$	$1/\text{s}$	40

Die Translation wird durch den Vektor $\mathbf{d}_{\text{icp}} \in \mathbb{R}^3$ dargestellt und die Rotation durch die Einheitsquaternion $\boldsymbol{\rho}_{\text{icp}} \in \mathbb{R}^4$ repräsentiert. Für die Beschreibung der Rotation durch Einheitsquaternionen und den Zusammenhängen zwischen der Rotationsmatrix und der Einheitsquaternion sei auf die Arbeit von Dam et al. [29] verwiesen. Der Eingangsvektor $\mathbf{u}_{\text{f}} \in \mathbb{R}^7$ besteht aus dem Vektor der Translation $\mathbf{d}_{\text{icp}}$ und der Einheitsquaternion $\boldsymbol{\rho}_{\text{icp}}$, $\mathbf{u}_{\text{f}} = \begin{bmatrix}\mathbf{d}_{\text{icp}}^{\text{T}} & \boldsymbol{\rho}_{\text{icp}}^{\text{T}}\end{bmatrix}^{\text{T}}$. Der i-te Eintrag von $\mathbf{u}_{\text{f}}$ wird mit u_{f}^i, $i = 1, 2, \ldots, 7$ bezeichnet. Die Filterung erfolgt mit sieben gleichwertigen Filtern, wobei jeder Eintrag vom Eingangsvektor u_{f}^i getrennt gefiltert wird. Für die Initialisierung der Filter wird das neutrale Element vom Eingangsvektor $\mathbf{u}_{\text{f},0}$ benötigt. Das neutrale Element der Translation und der Quaternion ist $\mathbf{d}_0 = \begin{bmatrix}0 & 0 & 0\end{bmatrix}^{\text{T}}$ sowie $\boldsymbol{\rho}_0 = \begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}^{\text{T}}$. Die verwendeten Filter haben mit den Zustandsvektoren $\boldsymbol{\chi}_{\text{f}}^i = \begin{bmatrix}x_{\text{f}}^i & \dot{x}_{\text{f}}^i\end{bmatrix}^{\text{T}}$ die Form

$$\dot{\boldsymbol{\chi}}_{\text{f}}^i = \mathbf{A}_{\text{f}}\boldsymbol{\chi}_{\text{f}}^i + \mathbf{b}\,u_{\text{f}}^i \tag{3.23a}$$

$$\text{mit} \quad \mathbf{A}_{\text{f}} = \begin{bmatrix} 0 & 1 \\ -b_{\text{f},0} & -b_{\text{f},1} \end{bmatrix} \tag{3.23b}$$

$$\mathbf{b}_{\text{f}} = \begin{bmatrix} 0 \\ b_{\text{f},0} \end{bmatrix} \tag{3.23c}$$

$$\text{und} \quad \boldsymbol{\chi}_{\text{f},0}^i = \begin{bmatrix} u_{\text{f},0}^i & 0 \end{bmatrix}^{\text{T}}. \tag{3.23d}$$

Die Parameter der Filter sind in Tabelle 3.2 zusammengefasst. Im Folgenden kennzeichnet der Überstrich ($\bar{}$) die stetige gefilterte Variante des Signals. Um eine Einheitsquaternion $\bar{\boldsymbol{\rho}}_{\text{icp}}$ und deren Ableitung $\dot{\bar{\boldsymbol{\rho}}}_{\text{icp}}$ zu erhalten, müssen die entsprechenden Zustände der Filter normiert werden. Für die Ableitung $\dot{\bar{\boldsymbol{\rho}}}_{\text{icp}}$ wird die Einheitsquaternion $\bar{\boldsymbol{\rho}}_{\text{icp}}$ analytisch abgeleitet. Die Normierung hat keinen Einfluss auf die Rotation, siehe [29]. Der Ausgang

berechnet sich damit zu

$$\bar{\mathbf{d}}_{\text{icp}} = \begin{bmatrix} x_{\text{f}}^1 \\ x_{\text{f}}^2 \\ x_{\text{f}}^3 \end{bmatrix} \tag{3.24a}$$

$$\dot{\bar{\mathbf{d}}}_{\text{icp}} = \begin{bmatrix} \dot{x}_{\text{f}}^1 \\ \dot{x}_{\text{f}}^2 \\ \dot{x}_{\text{f}}^3 \end{bmatrix} \tag{3.24b}$$

$$\bar{\boldsymbol{\rho}}_{\text{icp}} = \frac{1}{\sqrt{\left(x_{\text{f}}^4\right)^2 + \left(x_{\text{f}}^5\right)^2 + \left(x_{\text{f}}^6\right)^2 + \left(x_{\text{f}}^7\right)^2}} \begin{bmatrix} x_{\text{f}}^4 \\ x_{\text{f}}^5 \\ x_{\text{f}}^6 \\ x_{\text{f}}^7 \end{bmatrix} \tag{3.24c}$$

$$\begin{aligned} \dot{\bar{\boldsymbol{\rho}}}_{\text{icp}} = {} & \frac{1}{\sqrt{\left(x_{\text{f}}^4\right)^2 + \left(x_{\text{f}}^5\right)^2 + \left(x_{\text{f}}^6\right)^2 + \left(x_{\text{f}}^7\right)^2}} \begin{bmatrix} \dot{x}_{\text{f}}^4 \\ \dot{x}_{\text{f}}^5 \\ \dot{x}_{\text{f}}^6 \\ \dot{x}_{\text{f}}^7 \end{bmatrix} - \\ & \frac{x_{\text{f}}^4 \dot{x}_{\text{f}}^4 + x_{\text{f}}^5 \dot{x}_{\text{f}}^5 + x_{\text{f}}^6 \dot{x}_{\text{f}}^6 + x_{\text{f}}^7 \dot{x}_{\text{f}}^7}{\left(\sqrt{\left(x_{\text{f}}^4\right)^2 + \left(x_{\text{f}}^5\right)^2 + \left(x_{\text{f}}^6\right)^2 + \left(x_{\text{f}}^7\right)^2}\right)^3} \begin{bmatrix} x_{\text{f}}^4 \\ x_{\text{f}}^5 \\ x_{\text{f}}^6 \\ x_{\text{f}}^7 \end{bmatrix} . \end{aligned} \tag{3.24d}$$

Für die Regelung müssen die Ergebnisse der Filter wieder zusammengefasst bzw. umgewandelt werden. Die Winkelgeschwindigkeit $\bar{\boldsymbol{\omega}}_{\text{icp}}$ berechnet sich aus der Einheitsquaternion $\bar{\boldsymbol{\rho}}_{\text{icp}}$ und deren Ableitung $\dot{\bar{\boldsymbol{\rho}}}_{\text{icp}}$ mithilfe des Produktes zweier Quaternionen $\otimes$ sowie der komplexen Konjugation der Quaternion *, siehe [30]. Die Funktion $\text{Im}(\cdot)$ reduziert die Quaternion $\in \mathbb{R}^4$ auf deren Vektorteil $\in \mathbb{R}^3$. Die benötigten gefilterten Signale für die Regelung ergeben sich gemäß

$$\bar{\mathbf{H}}_{\text{icp}} = \begin{bmatrix} \bar{\mathbf{R}}_{\text{icp}}\left(\bar{\boldsymbol{\rho}}_{\text{icp}}\right) & \bar{\mathbf{d}}_{\text{icp}} \\ \mathbf{0} & 1 \end{bmatrix} \tag{3.25a}$$

$$\dot{\bar{\mathbf{d}}}_{\text{icp}} = \begin{bmatrix} \dot{x}_{\text{f}}^1 \\ \dot{x}_{\text{f}}^2 \\ \dot{x}_{\text{f}}^3 \end{bmatrix} \tag{3.25b}$$

$$\bar{\boldsymbol{\omega}}_{\text{icp}} = 2\ \text{Im}\left(\dot{\bar{\boldsymbol{\rho}}}_{\text{icp}} \otimes \left(\bar{\boldsymbol{\rho}}_{\text{icp}}\right)^*\right) . \tag{3.25c}$$

Für die Implementierung müssen die zeitkontinuierlichen Filter diskretisiert werden, wofür ein Zero-Order-Hold-Glied mit der Abtastzeit des Reglers T_{R} verwendet wird, siehe [31]. Schließlich lauten die zeitdiskreten Filter

$$\boldsymbol{\chi}_{\text{f},k+1}^i = \boldsymbol{\Phi}_{\text{f}} \boldsymbol{\chi}_{\text{f},k}^i + \boldsymbol{\Gamma}\, u_{\text{f},k}^i \tag{3.26a}$$

$$\text{mit} \quad \boldsymbol{\Phi}_{\text{f}} = \exp(\mathbf{A}_{\text{f}} T_{\text{R}}) \tag{3.26b}$$

$$\boldsymbol{\Gamma}_{\text{f}} = \int_0^{T_{\text{R}}} \exp(\mathbf{A}_{\text{f}} t)\, \text{d}t\, \mathbf{b}_{\text{f}} \tag{3.26c}$$

3.3.3 Verwenden der vorherigen Posenbestimmung

Im Ablauf welcher in Abschnitt 3.2.5 beschrieben ist, ist jeder Durchgang der Posenbestimmung unabhängig vom vorherigen Durchgang. Um das Ergebnis der vorherigen Posenbestimmung $\mathbf{H}_{\text{icp}}^{\epsilon}$ bei der aktuellen Posenbestimmung mitzuberücksichtigen, wird die iterierende Punktwolke P_0 auf

$$P_0 = f_{\text{H}}\left(\mathbf{H}_{\text{icp}}^{\epsilon}, P_{\text{U}}\right) \tag{3.27}$$

initialisiert. Unter der Annahme, dass zwischen den Abtastzeiten der Posenbestimmung die Änderung der Pose klein ist, reduziert sich dadurch der Rechenaufwand des Algorithmus. Das Ergebnis der Posenbestimmung $\mathbf{H}_{\text{icp}}$ muss gemäß

$$\mathbf{H}_{\text{icp}} = \mathbf{H}_{\text{icp}}^{\text{v}}\,\mathbf{H}_{\text{icp}}^{\epsilon} \tag{3.28}$$

angepasst werden. $\mathbf{H}_{\text{icp}}^{\text{v}}$ bezeichnet dabei das vorläufige Ergebnis des Algorithmus.

3.4 Vergleich verschiedener ICP-Algorithmen

In diesem Abschnitt wird der hergeleitete ApPzE-ICP-Algorithmus aus Abschnitt 3.2.5 mit zwei anderen ICP-Algorithmen in einer Simulation ohne Industrieroboter auf die Eignung für die Posenbestimmung getestet. Es wird die Genauigkeit und die Laufzeit der verschiedenen Algorithmen bestimmt und verglichen. Dafür werden zuerst in Abschnitt 3.4.1 und Abschnitt 3.4.2 der Aufbau sowie der Ablauf der Simulation erklärt und danach in Abschnitt 3.4.3 die Ergebnisse diskutiert.

3.4.1 Aufbau der Simulation

Das Ziel ist es die Eignung der verschiedenen Algorithmen für die Posenbestimmung zu testen. Dafür werden zwei Eigenschaften der Algorithmen untersucht:

- Der systematischer Fehler der Algorithmen.
- Die Genauigkeit der Posenbestimmung der Algorithmen.

In Abbildung 3.5 ist der Aufbau der Simulation bestehend aus einem 2D-Lasertriangulationssensor und dem Objekt dargestellt. Die Simulation erfolgt ohne Industrieroboter und die Sensorpose $\mathcal{S}$ und die Objektpose $\mathcal{O}$ werden frei im Raum platziert. Es werden im Folgenden drei ICP-Algorithmen verglichen:

- PzP-ICP-Algorithmus mit exakter Berechnung der nächstgelegenen Punkte $P_{\text{Z},k}$, siehe [26] (abgekürzt mit ExPzP).
- ApPzE-ICP-Algorithmus mit exakter Berechnung der nächstgelegenen Punkte $P_{\text{Z},k}$, siehe [25, 26] (abgekürzt mit ExPzE).
- ApPzE-ICP-Algorithmus mit Approximation der Berechnung der nächstgelegenen Punkte $P_{\text{Z},k}$, siehe Abschnitt 3.2 (abgekürzt mit ApPzE).

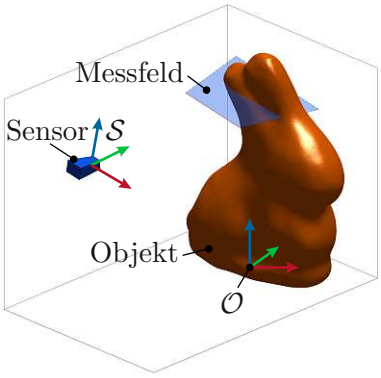


Abbildung 3.5: Darstellung der Simulation mit dem Objekt und einem 2D-Lasertriangulationssensor mit eingezeichnetem Messfeld.

3.4.2 Ablauf der Simulation

Zunächst werden die Messpunkte des 2D-Lasertriangulationssensors ${}_{\mathcal{S}}P_{\mathrm{S}}$, wie in Abschnitt 2.3.3 beschrieben, berechnet. Die Messpunkte werden mithilfe der homogenen Transformation

$$ {}_{\mathcal{S}}\mathbf{H}_{\mathrm{V}} = \mathbf{H}_{\mathrm{Tz},d_{\mathrm{v}}}\,\mathbf{H}_{\mathrm{Ry},\alpha_{\mathrm{v}}} \tag{3.29} $$

versetzt, wobei der Versatz aus einer translatorischen Verschiebung d_{v} entlang der z-Achse und einer Rotation α_{v} um die y-Achse des Sensorkoordinatensystems $\mathcal{S}$ besteht. Dies simuliert eine mögliche Ungenauigkeit bei der Positionierung des Objektes bzw. des Sensors in einer Roboteranwendung. Die versetzten Messpunkte ergeben sich in der Form

$$ {}_{\mathcal{S}}P_{\mathrm{V}} = f_{\mathrm{H}}({}_{\mathcal{S}}\mathbf{H}_{\mathrm{V}}, {}_{\mathcal{S}}P_{\mathrm{S}}) \ . \tag{3.30} $$

Für den ICP-Algorithmus werden die Messpunkte relativ zum Objektkoordinatensystem $\mathcal{O}$ mit

$$ {}_{\mathcal{O}}P_{\mathrm{V}} = f_{\mathrm{H}}\Big({}^{\mathcal{S}}_{\mathcal{O}}\mathbf{H}, {}_{\mathcal{S}}P_{\mathrm{V}}\Big) \tag{3.31} $$

berechnet. Der ICP-Algorithmus bestimmt die homogene Transformation $\mathbf{H}_{\mathrm{icp}}$, welche einer Schätzung der inversen Verschiebung ${}_{\mathcal{O}}\hat{\mathbf{H}}_{\mathrm{V}}{}^{-1}$ in Bezug auf das Objektkoordinatensystem $\mathcal{O}$ entspricht, d. h. es gilt

$$ {}_{\mathcal{O}}\hat{P}_{\mathrm{S}} = f_{\mathrm{H}}(\mathbf{H}_{\mathrm{icp}}, {}_{\mathcal{O}}P_{\mathrm{V}}) \ . \tag{3.32} $$

Als Fehlermetriken werden die translatorische Abweichung (3.17) und die rotatorische Abweichung (3.18) zwischen den ursprünglichen Messpunkten ${}_{\mathcal{O}}P_{\mathrm{S}}$ und der Lösung des ICP-Algorithmus ${}_{\mathcal{O}}\hat{P}_{\mathrm{S}}$ gemäß

$$ \mathbf{r} = \begin{bmatrix} r_{\mathrm{T}} \\ r_{\mathrm{R}} \end{bmatrix} = \begin{bmatrix} f_{\mathrm{E,T}}(\mathbf{H}_{\mathrm{E}}, {}_{\mathcal{O}}P_{\mathrm{S}}) \\ f_{\mathrm{E,R}}(\mathbf{H}_{\mathrm{E}}) \end{bmatrix} \tag{3.33} $$

verwendet. Die homogene Transformation $\mathbf{H}_{\mathrm{E}}$ von ${}_{\mathcal{O}}P_{\mathrm{S}}$ nach ${}_{\mathcal{O}}\hat{P}_{\mathrm{S}}$ ergibt sich durch

$$ \mathbf{H}_{\mathrm{E}} = {}^{\mathcal{S}}_{\mathcal{O}}\mathbf{H}\,{}_{\mathcal{S}}\mathbf{H}_{\mathrm{V}}\,{}^{\mathcal{O}}_{\mathcal{S}}\mathbf{H}\,\mathbf{H}_{\mathrm{icp}} \ . \tag{3.34} $$

Tabelle 3.3: Parameter für den Vergleich der ICP-Algorithmen.

Bezeichnung	Symbol	Einheit	Wert	
Verschiebung Translation	d_v	mm	-5	
Verschiebung Rotation	α_v	°	2	
maximale Iteration	k_max	1	6	
Toleranz Translation	$\tau_\mathrm{T,min}$	µm	100	
Toleranz Rotation	$\tau_\mathrm{R,min}$	°	0,1	
maximale Translation	$\tau_\mathrm{T,max}$	mm	50	
maximale Rotation	$\tau_\mathrm{R,max}$	°	5	
Anzahl Dreiecke	N_D	1	1	$\cdot 10^5$
Oberfläche Objekt		m^2	0,107	
Höhe Objekt		mm	252	
Breite Objekt		mm	230	
Tiefe Objekt		mm	112	

Die Genauigkeit des ICP-Algorithmus variiert im Allgemeinen für verschiedene Positionen. Die Genauigkeit hängt von den Messpunkten und der entsprechenden lokalen Freiformoberfläche des Modells des Objektes ab. Um diese Variation in der Simulation zu berücksichtigen, werden für jeden ICP-Algorithmus zehn verschiedene Sensorposen simuliert. Für die Untersuchung des systematischen Fehlers erfolgt die Simulation ohne Versatz ${}_\mathcal{S}\mathbf{H}_\mathrm{V} = \mathbf{E}$, wodurch die Abweichung (3.33) dem systematischen Fehler des Algorithmus entspricht. Für die Untersuchung der Genauigkeit der Posenbestimmung erfolgt die Simulation mit Versatz gemäß (3.29).

Die gewählten Parameter für die Simulation sowie die geometrischen Eigenschaften des Objektes sind in Tabelle 3.3 aufgelistet.

3.4.3 Ergebnisse

Die Ergebnisse der Simulation sind in Tabelle 3.4 und Tabelle 3.5 zusammengefasst. Für die Bewertung wird die Fehlermetrik (3.33) von zehn Posen gemittelt (mean) und die Standardabweichung (std) berechnet. Weiters wird die durchschnittliche Anzahl der benötigten Iterationen und die durchschnittliche Laufzeit der Algorithmen für die Bewertung der Performance notiert. Im Folgenden werden die Ergebnisse ohne und mit Verschiebung ${}_\mathcal{S}\mathbf{H}_\mathrm{V}$ diskutiert.

Ohne Verschiebung

In Tabelle 3.4 sind die Ergebnisse der Simulation ohne Verschiebung zusammengefasst. Da die Messpunkte nicht verschoben werden, ist das Ergebnis des Algorithmus im Idealfall die Einheitsmatrix und die translatorische Abweichung r_T sowie die rotatorische Abweichung r_R sind 0.

Die Algorithmen ExPzP und ExPzE berechnen die nächstgelegenen Punkte exakt auf der diskreten Oberfläche und haben aus diesem Grund nur einen vernachlässigbar

Tabelle 3.4: Ergebnis der Posenbestimmung ohne Verschiebung.

	Name	Laufzeit	Iterationen	Translation		Rotation	
	Symbol	mean t_L	mean k_{it}	mean r_T	std r_T	mean r_R	std r_R
	Einheit	s	1	m	m	°	°
ExPzP		21	1	$2 \cdot 10^{-17}$	$1 \cdot 10^{-17}$	$4 \cdot 10^{-7}$	$8 \cdot 10^{-7}$
ExPzE		19	1	$7 \cdot 10^{-17}$	$7 \cdot 10^{-17}$	$6 \cdot 10^{-7}$	$6 \cdot 10^{-7}$
ApPzE		$8 \cdot 10^{-3}$	1	$4{,}9 \cdot 10^{-6}$	$3{,}1 \cdot 10^{-6}$	$27 \cdot 10^{-3}$	$26 \cdot 10^{-3}$

Tabelle 3.5: Ergebnis der Posenbestimmung mit Verschiebung, ExPzP-Algorithmus hat bei 2 von 10 Posen die maximale Translation $\tau_{T,max}$ oder maximale Rotation $\tau_{R,max}$ überschritten.

	Name	Laufzeit	Iterationen	Translation		Rotation	
	Symbol	mean t_L	mean k_{it}	mean r_T	std r_T	mean r_R	std r_R
	Einheit	s	1	m	m	°	°
ExPzP		354	15,4	$4 \cdot 10^{-3}$	$489 \cdot 10^{-6}$	4,4	1,2
ExPzE		116	5,5	$12 \cdot 10^{-9}$	$22 \cdot 10^{-9}$	$45 \cdot 10^{-6}$	$93 \cdot 10^{-6}$
ApPzE		$23 \cdot 10^{-3}$	5,7	$4{,}5 \cdot 10^{-6}$	$2{,}4 \cdot 10^{-6}$	$26 \cdot 10^{-3}$	$26 \cdot 10^{-3}$

kleinen systematischen Fehler. Dieser entsteht aufgrund der begrenzten Genauigkeit der Numerik der Simulation. Der Algorithmus ApPzE berechnet die nächstgelegenen Punkte durch eine Approximation, siehe Abschnitt 3.2.3. Durch diese Näherung entsteht ein systematischer Fehler mit einer mittleren translatorischen Abweichung von 4,9 µm mit einer Standardabweichung von 3,1 µm. Die rotatorische Abweichung beträgt 27 Milligrad mit einer Standardabweichung von 26 Milligrad.

Alle drei Algorithmen benötigen eine einzige Iteration. Bei der Laufzeit gibt es große Unterschiede. Durch die Näherung und den Optimierungen benötigt der ApPzE-Algorithmus im Mittel 8 ms für eine Posenbestimmung während die Algorithmen ExPzE sowie ExPzP 19 s bzw. 21 s erfordern.

Mit Verschiebung

In Tabelle 3.5 sind die Ergebnisse der Simulation mit Verschiebung ${}_{\mathcal{S}}\mathbf{H}_V \neq \mathbf{E}$ dargestellt. Der Algorithmus ExPzP hat bei zwei von zehn Sensorposen die maximale Translation $\tau_{T,max}$ oder die maximale Rotation $\tau_{R,max}$ überschritten. Diese zwei Posen werden für die Berechnung des Mittelwertes und der Standardabweichung nicht miteinbezogen. Der translatorische Fehler für den ExPzP-Algorithmus ist im Mittel 4 mm und der rotatorische Fehler liegt bei 4,4°. Die Verschiebung der Messpunkte ist in der selben Größenordnung wie die Genauigkeit der Posenbestimmung. Daraus wird geschlossen, dass dieser Algorithmus nicht für diese Art von Anwendung geeignet ist. Der ExPzE-Algorithmus hat eine sehr hohe Genauigkeit mit einen mittleren Fehler von 12 nm und 45 Mikrograd. Der Algorithmus ApPzE erreicht in etwa die selbe Genauigkeit wie im vorherigen Experiment

und benötigt mit 5,7 Iterationen ähnlich viele Iterationen wie ExPzE-Algorithmus mit 5,5. Der Unterschied in der Laufzeit ist bei den Algorithmen wiederum sehr groß, mit 23 ms für den ApPzE-Algorithmus und 116 s für ExPzE-Algorithmus.

Diskussion

Schließlich werden die Ergebnisse aus Tabelle 3.5 in Relation zu kommerziellen Industrierobotern gesetzt. In dem Datenblatt des KUKA LBR iiwa 14 R820 [32] wird eine Wiederholgenauigkeit von ±100 µm angegeben. Die Absolutgenauigkeit des Industrieroboters ist in diesem Datenblatt nicht angegeben und ist im Allgemeinen aber deutlich größer als die Wiederholgenauigkeit. Der ApPzE-Algorithmus ist daher um mehrere Größenordnungen genauer als die Wiederholgenauigkeit des Industrieroboters und um mehrere Größenordnungen schneller als der ExPzE-Algorithmus. Aus diesen Gründen wird im weiteren Verlauf dieser Arbeit der ApPzE-Algorithmus verwendet. Der systematische Fehler desApPzE-Algorithmus ist auch vernachlässigbar.

Der Grund für den großen Geschwindigkeitsunterschied zwischen dem ApPzE-Algorithmus und den ExPzE-Algorithmus ist, dass beim ApPzE-Algorithmus die korrespondierenden Punkte durch eine Approximation berechnet werden, siehe Abschnitt 3.2.3.

4 Trajektorienplanung und Regelung

In diesem Kapitel werden die Trajektorienplanung und die Regelung des Industrieroboters in der Simulation beschrieben. Die Planung des Pfades und der Trajektorie wird in Abschnitt 4.1 vorgestellt. Die Regelung mithilfe der inversen Dynamik und die Verwendung der zusätzlichen Information über die Pose aus der Messung des 2D-Lasertriangulationssensors werden in Abschnitt 4.2 erläutert.

4.1 Pfad- und Trajektorienplanung

In dieser Arbeit soll in einer Simulation ein Werkstück entlang einer Trajektorie auf der Werkstückoberfläche von einem Werkzeug poliert werden. Dafür wird in diesem Abschnitt die Erstellung der Trajektorie für die Regelung des Industrieroboters beschrieben. In Abschnitt 4.1.1 wird zuerst der benötigte Pfad charakterisiert. Die Umwandlung vom Pfad auf die gewünschte Trajektorie wird in Abschnitt 4.1.2 erläutert.

4.1.1 Pfadplanung

Die folgenden Ausführungen basieren auf der Arbeit von Hartl-Nesic [33] und fassen die wichtigsten Gleichungen zusammen. Ein parametrierter Pfad $\boldsymbol{\pi}(p) : I \subseteq \mathbb{R} \mapsto \mathbb{R}^3$ gibt die Pfadposition in Abhängigkeit des Pfadparameters $p \in I$ an. In dieser Arbeit wird angenommen, dass Pfade nach der Pfadlänge parametriert werden, d. h. es gilt $\frac{\partial \boldsymbol{\pi}(p)}{\partial p} = 1 \forall p \in I$.

Ein Pfad $\boldsymbol{\pi}(p)$ ist durch die Reihe von Punkten auf der Oberfläche definiert, welche durch B-Splines interpoliert werden. In jedem Punkt wird ein Kontaktkoordinatensystem $\mathcal{K}$ relativ zum Objektkoordinatensystem $\mathcal{O}$ gemäß

$$
{}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(p) = \begin{bmatrix} \frac{\partial \boldsymbol{\pi}(p)}{\partial p} & \frac{\partial \boldsymbol{\pi}(p)}{\partial p} \times \mathbf{n}(p) & \mathbf{n}(p) & \boldsymbol{\pi}(p) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.1}
$$

definiert. Die x-Achse des Kontaktkoordinatensystems $\mathcal{K}$ ist entlang des Pfades orientiert, die z-Achse entspricht dem Normalvektor der Oberfläche $\mathbf{n}(p)$ in diesem Punkt. Die y-Achse ist derart gewählt, dass sich ein rechtshändiges Koordinatensystem ergibt. Für die Erstellung der Trajektorie für die Regelung ist ein dreifach stetig differenzierbarer Pfad $\mathcal{C}^3$ von Nöten. Damit ergibt sich der Pfad samt lokaler Orientierung und dessen Ableitungen entlang des Pfades zu ${}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(p)$, $\frac{\partial}{\partial p}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(p)\right)$, $\frac{\partial^2}{\partial p^2}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(p)\right)$. In Abbildung 4.1 ist ein generierter Pfad am Objekt dargestellt.

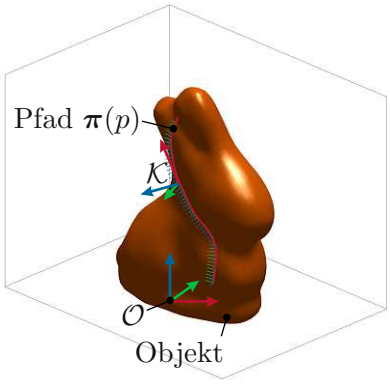


Abbildung 4.1: Darstellung eines Pfades $\boldsymbol{\pi}(p)$ am Objekt mit Kontaktkoordinatensystem $\mathcal{K}$.

4.1.2 Trajektorienplanung

Der in Abschnitt 4.1.1 erstellte Pfad muss für die Regelung in eine Trajektorie als Zeitfunktion umgewandelt werden. Der Pfad soll dabei mit einer konstanten Geschwindigkeit v_d abgefahren werden. Der Pfadgeschwindigkeit v_d des nach der Pfadlänge parametrierten Pfades entspricht der Zeitableitung des Pfadparameters $\frac{\partial p(t)}{\partial t} = v_\mathrm{d}$.

Es wird für den Beginn der Simulation eine Ruhezeit T_init eingeführt. Die Endzeit T_end für die Bewegung des Roboters ergibt sich aus der Länge des Pfades l_p zu

$$T_\mathrm{end} = T_\mathrm{init} + \frac{l_\mathrm{p}}{v_\mathrm{d}} \,. \tag{4.2}$$

Der Pfadparameter berechnet sich mit einem linearen Verlauf zu

$$p(t) = \begin{cases} 0, & \text{wenn} \quad t \leq T_\mathrm{init} \\ v_\mathrm{d}(t - T_\mathrm{init}), & \text{wenn} \quad T_\mathrm{init} < t \leq T_\mathrm{end} \\ l_\mathrm{p}, & \text{sonst} \end{cases} \,. \tag{4.3}$$

Damit die Trajektorie zweifach stetig differenzierbar $\mathcal{C}^2$ ist, muss der Pfadparameter $p(t)$ mit einem Filter geglättet werden. Der gefilterte Pfadparameter wird mit $\bar{p}$ bezeichnet. Der Filter mit dem Zustandsvektor $\boldsymbol{\chi}_\mathrm{p} = \begin{bmatrix} \bar{p} & \dot{\bar{p}} & \ddot{\bar{p}} \end{bmatrix}^\mathrm{T}$ berechnet sich zu

$$\dot{\boldsymbol{\chi}}_\mathrm{p} = \mathbf{A}_\mathrm{p}\boldsymbol{\chi}_\mathrm{p} + \mathbf{b}_\mathrm{p} p \tag{4.4a}$$

$$\text{mit} \quad \mathbf{A}_\mathrm{p} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -b_{\mathrm{p},0} & -b_{\mathrm{p},1} & -b_{\mathrm{p},2} \end{bmatrix} \tag{4.4b}$$

$$\mathbf{b}_\mathrm{p} = \begin{bmatrix} 0 \\ 0 \\ b_{\mathrm{p},0} \end{bmatrix} \,. \tag{4.4c}$$

Der Zustand $\boldsymbol{\chi}_\mathrm{p}$ entspricht dabei gleichzeitig dem Ausgang des Filters. Die Parameter des Pfades und des Filters sind in Tabelle 4.1 zusammengefasst. Für die Implementierung muss

Tabelle 4.1: Parameter des Pfades und des Filters.

Symbol	Einheit	Wert
T_{init}	s	1
$b_{\text{p},0}$	$1/\text{s}^3$	1000
$b_{\text{p},1}$	$1/\text{s}^2$	300
$b_{\text{p},2}$	$1/\text{s}$	30

der zeitkontinuierliche Filter diskretisiert werden. Dafür wird ein Zero-Order-Hold-Glied mit der Abtastzeit des Reglers T_R verwendet, siehe [31]. Der Filter ergibt sich zu

$$\boldsymbol{\chi}_{\text{p},k+1} = \boldsymbol{\Phi}_\text{p}\boldsymbol{\chi}_{\text{p},k} + \boldsymbol{\Gamma}_\text{p} p_k \tag{4.5a}$$

$$\text{mit} \quad \boldsymbol{\Phi}_\text{p} = \exp(\mathbf{A}_\text{p} T_\text{R}) \tag{4.5b}$$

$$\boldsymbol{\Gamma}_\text{p} = \int_0^{T_\text{R}} \exp(\mathbf{A}_\text{p} t)\,\mathrm{d}t\,\mathbf{b}_\text{p}\ . \tag{4.5c}$$

Die Zeitableitungen der vorgegebenen Trajektorie ${}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(\bar{p})$ werden schließlich mithilfe der Zeitableitungen des gefilterten Pfadparameters und der Kettenregel gemäß

$${}^{\mathcal{K}}_{\mathcal{O}}\dot{\mathbf{H}}(\bar{p},\dot{\bar{p}}) = \frac{\partial}{\partial\bar{p}}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(\bar{p})\right)\dot{\bar{p}} \tag{4.6a}$$

$${}^{\mathcal{K}}_{\mathcal{O}}\ddot{\mathbf{H}}(\bar{p},\dot{\bar{p}},\ddot{\bar{p}}) = \frac{\partial^2}{\partial\bar{p}^2}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(\bar{p})\right)\dot{\bar{p}}^2 + \frac{\partial}{\partial\bar{p}}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}(\bar{p})\right)\ddot{\bar{p}} \tag{4.6b}$$

berechnet.

Das Objekt soll durch das Werkzeug entlang des vorgegebenen Pfades poliert werden. Damit das Werkzeug mit dem Werkzeugkoordinatensystem $\mathcal{T}$ während der Bearbeitung im Kontakt mit dem Werkstück bleibt und das Werkzeug für den Polierprozess normal zur Oberfläche steht, wird die homogene Transformation ${}^{\mathcal{T}}_{\mathcal{K}}\mathbf{H} = \mathbf{E}$ gesetzt. Schließlich berechnet sich die vorgegebene Trajektorie (Index „d“ für *desired*) mithilfe der Funktion $f_\text{US}(\cdot)$ aus (2.21) zu

$${}^{\mathcal{T}}_{\mathcal{O}}\mathbf{d}_\text{d} = f_\text{d}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}\right) \tag{4.7a}$$

$${}^{\mathcal{T}}_{\mathcal{O}}\mathbf{R}_\text{d} = {}^{\mathcal{K}}_{\mathcal{O}}\mathbf{R} \tag{4.7b}$$

$${}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{y}}_\text{d} = \begin{bmatrix} {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{d}}_\text{d} \\ {}^{\mathcal{T}}_{\mathcal{O}}\boldsymbol{\omega}_\text{d} \end{bmatrix} = \begin{bmatrix} f_\text{d}\left({}^{\mathcal{K}}_{\mathcal{O}}\dot{\mathbf{H}}\right) \\ f_\text{US}\left({}^{\mathcal{K}}_{\mathcal{O}}\dot{\mathbf{R}}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{R}\right)^\text{T}\right) \end{bmatrix} \tag{4.7c}$$

$${}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{y}}_\text{d} = \begin{bmatrix} {}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{d}}_\text{d} \\ {}^{\mathcal{T}}_{\mathcal{O}}\dot{\boldsymbol{\omega}}_\text{d} \end{bmatrix} = \begin{bmatrix} f_\text{d}\left({}^{\mathcal{K}}_{\mathcal{O}}\ddot{\mathbf{H}}\right) \\ f_\text{US}\left({}^{\mathcal{K}}_{\mathcal{O}}\ddot{\mathbf{R}}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{R}\right)^\text{T} + {}^{\mathcal{K}}_{\mathcal{O}}\dot{\mathbf{R}}\left({}^{\mathcal{K}}_{\mathcal{O}}\dot{\mathbf{R}}\right)^\text{T}\right) \end{bmatrix} \tag{4.7d}$$

mit der Rotationsmatrix ${}^{\mathcal{K}}_{\mathcal{O}}\mathbf{R} = f_{\mathrm{R}}\left({}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}\right)$ und deren Ableitungen

$$ {}^{\mathcal{K}}_{\mathcal{O}}\dot{\mathbf{R}} = f_{\mathrm{R}}\left({}^{\mathcal{K}}_{\mathcal{O}}\dot{\mathbf{H}}\right) \tag{4.8a} $$

$$ {}^{\mathcal{K}}_{\mathcal{O}}\ddot{\mathbf{R}} = f_{\mathrm{R}}\left({}^{\mathcal{K}}_{\mathcal{O}}\ddot{\mathbf{H}}\right) . \tag{4.8b} $$

4.2 Regelung

Im folgenden Abschnitt wird die Regelung des Industrieroboters KUKA LBR iiwa 14 R820 behandelt. Der Entwurf der Regelung erfolgt auf Basis des nominellen Modells des Roboters. In Abschnitt 4.2.1 wird die Kinematik für ortsfeste Werkzeuge eingeführt und hergeleitet. Die Regelung des Roboters erfolgt mithilfe der inversen Dynamik, welche in Abschnitt 4.2.2 erläutert wird. Die zusätzliche Information über die Pose des Objektes durch den modifizierten ICP-Algorithmus aus Kapitel 3 wird in Abschnitt 4.2.3 mit den Ausgang des nominellen Modells vereint. In Abschnitt 4.2.4 wird die Trajektorienfolgeregelung hergeleitet. Für die Stabilisierung des Nullraumes wird in Abschnitt 4.2.5 ein geeigneter Nullraumregler eingeführt.

4.2.1 Kinematik für ortsfeste Werkzeuge

Wie in Abschnitt 4.1 beschrieben wird, erfolgt die Trajektorienplanung entlang der Oberfläche des Objektes im Objektkoordinatensystem $\mathcal{O}$. In Abbildung 4.2 ist der Aufbau der Simulation dargestellt. Das Ziel der Regelung ist es, mit dem Industrieroboter das Objekt entlang der Trajektorie im Kontakt mit dem Werkzeug zu führen, um dieses dadurch zu polieren. Das Werkzeug ist dabei ortsfest und dessen Pose wird durch das Werkzeugkoordinatensystem $\mathcal{T}$ beschrieben. Die z-Achse ist in Richtung des Werkzeugs orientiert und die x-Achse parallel zur z-Achse des Weltkoordinatensystems $\mathcal{W}$. Um die Kinematik des feststehenden Werkzeugs zu berücksichtigen, wird die direkte Kinematik als homogene Transformation ${}^{\mathcal{T}}_{\mathcal{O}}\mathbf{H}(\mathbf{q})$, d. h. als Transformation des Werkzeugkoordinatensystems $\mathcal{T}$ in Bezug auf das Objektkoordinatensystems $\mathcal{O}$ in der Form

$$ {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{H}(\mathbf{q}) = {}^{\mathcal{E}}_{\mathcal{O}}\mathbf{H}\,{}^{\mathcal{B}}_{\mathcal{E}}\mathbf{H}(\mathbf{q})\,{}^{\mathcal{W}}_{\mathcal{B}}\mathbf{H}\,{}^{\mathcal{T}}_{\mathcal{W}}\mathbf{H} \tag{4.9} $$

berechnet. Für den Versuchsaufbau in Abbildung 4.2 ist ${}^{\mathcal{E}}_{\mathcal{O}}\mathbf{H} = \mathbf{E}$, da $\mathcal{E} = \mathcal{O}$ gilt und ${}^{\mathcal{B}}_{\mathcal{E}}\mathbf{H}(\mathbf{q})$ ist die Inverse der direkten Kinematik des Roboters (2.16). Die direkte Kinematik (4.9) wird als Systemausgang des Roboters festgelegt, das entspricht der Pose des Werkzeugkoordinatensystem $\mathcal{T}$ relativ zum Objektkoordinatensystem $\mathcal{O}$.

Für die Regelung werden auch die entsprechende translatorische Geschwindigkeit ${}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{d}}$ und die Winkelgeschwindigkeit ${}^{\mathcal{T}}_{\mathcal{O}}\boldsymbol{\omega}$ benötigt. Die Berechnung erfolgt analog zu Abschnitt 2.2.2. Die Zeitableitungen des Systemausgangs ${}^{\mathcal{T}}_{\mathcal{O}}\mathbf{y}$ ergeben

$$ {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{y}} = \begin{bmatrix} {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{d}} \\ {}^{\mathcal{T}}_{\mathcal{O}}\boldsymbol{\omega} \end{bmatrix} = {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{4.10a} $$

$$ {}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{y}} = {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{J}}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{J}(\mathbf{q})\ddot{\mathbf{q}} . \tag{4.10b} $$

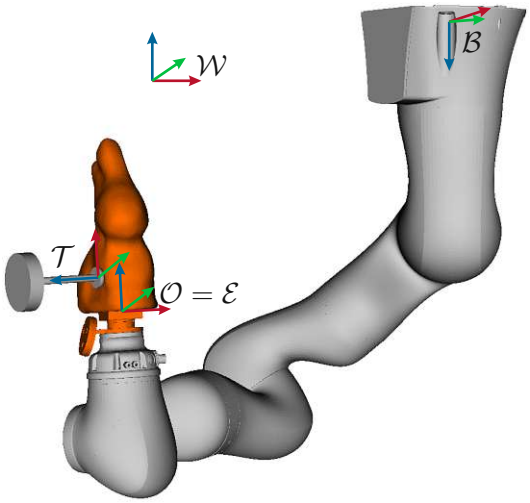


Abbildung 4.2: Darstellung der Simulation mit Werkzeugkoordinatensystem $\mathcal{T}$, nomineller Objektpose $\mathcal{O}$, nomineller Endeffektorpose $\mathcal{E}$, Basiskoordinatensystem des Roboters $\mathcal{B}$ und Weltkoordinatensystem $\mathcal{W}$.

4.2.2 Inverse Dynamik

Die Regelung des KUKA LBR iiwa 14 R820 Industrieroboters erfolgt mithilfe der inversen Dynamik. Dabei handelt es sich um eine Eingangs-Ausgangslinearisierung für vollaktuierte Starrkörpersysteme, siehe [18]. Der Entwurf der Regelung erfolgt auf Basis des nominellen Modells des Roboters. Der Industrieroboter KUKA LBR iiwa 14 R820 ist ein redundanter siebenachsiger Roboter mit den sieben Drehmomenten $\boldsymbol{\tau}$ der Motoren als Stellgröße. Die Drehmomente

$$\boldsymbol{\tau} = \boldsymbol{\tau}_\mathrm{R} + \boldsymbol{\tau}_\mathrm{N} \tag{4.11}$$

werden auf die Drehmomente des Reglers des Arbeitsraumes $\boldsymbol{\tau}_\mathrm{R}$ und die Drehmomente des Nullraumreglers $\boldsymbol{\tau}_\mathrm{N}$ geteilt. Mithilfe des Reglers für den Arbeitsraum werden die sechs Freiheitsgrade des dreidimensionalen Raumes stabilisiert. Die Regelung des Starrkörpersystems (2.24) erfolgt mit der inversen Dynamik gemäß

$$\boldsymbol{\tau}_\mathrm{R} = \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) + \mathbf{M}(\mathbf{q})\, {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{J}^{\dagger}(\mathbf{q})\left(\mathbf{u} - {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{J}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}\right), \tag{4.12}$$

welche auf das lineare Eingangs-Ausgangsverhalten mit der neuen Stellgröße $\mathbf{u}$ in der Form

$${}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{y}} = \mathbf{u} \tag{4.13}$$

führt. Die Drehmomente des Nullraums $\boldsymbol{\tau}_\mathrm{N}$ regeln zusätzlich den Nullraum, siehe Abschnitt 4.2.5. In (4.12) wird die rechte Pseudoinverse $\mathbf{A}^{\dagger} = \mathbf{A}^{\mathrm{T}}\left(\mathbf{A}\mathbf{A}^{\mathrm{T}}\right)^{-1}$ einer nicht quadratischen Matrix $\mathbf{A}$, bei der die Zeilen linear unabhängig sind, verwendet.

4.2.3 Sensorfusion

Das Ziel der Sensorfusion ist es, den Systemausgang des nominellen Modells des Roboters (4.9), (4.10) mit der Posenbestimmung (3.25) zu kombinieren. Durch die Erweiterung kann

die Schätzung der Pose des Objektes $\hat{\mathcal{O}}$ für die Regelung verwendet werden. Für die Regelung wird die Schätzung der direkten Kinematik ${}^{\mathcal{T}}_{\hat{\mathcal{O}}}\mathbf{H}$, der translatorischen Geschwindigkeit ${}^{\mathcal{T}}_{\hat{\mathcal{O}}}\dot{\mathbf{d}}$ und der Winkelgeschwindigkeit ${}^{\mathcal{T}}_{\hat{\mathcal{O}}}\boldsymbol{\omega}$ benötigt.

Das Ergebnis der Posenbestimmung ist die homogene Transformation der geschätzten Objektpose $\hat{\mathcal{O}}$ relativ zur nominellen Objektpose $\mathcal{O}$. Aus diesem Grund ergibt sich das Ergebnis der Posenbestimmung (3.25) zu

$$ {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H} = \bar{\mathbf{H}}_{\text{icp}} \tag{4.14a} $$

$$ {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\dot{\mathbf{d}} = \dot{\bar{\mathbf{d}}}_{\text{icp}} \tag{4.14b} $$

$$ {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\boldsymbol{\omega} = \bar{\boldsymbol{\omega}}_{\text{icp}} \ . \tag{4.14c} $$

Die Erweiterung der direkten Kinematik (4.9) ergibt sich mit (4.14) zu

$$ {}^{\mathcal{T}}_{\hat{\mathcal{O}}}\mathbf{H}(\mathbf{q}) = {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H}\, {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{H}(\mathbf{q}) \ . \tag{4.15} $$

Die zeitliche Ableitung der Rotationsmatrix ${}^{\mathcal{O}}_{\hat{\mathcal{O}}}\dot{\mathbf{R}}$ berechnet sich gemäß

$$ {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\dot{\mathbf{R}} = f_{\text{S}}\left({}^{\mathcal{O}}_{\hat{\mathcal{O}}}\boldsymbol{\omega}\right) f_{\text{R}}\left({}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H}\right) \tag{4.16} $$

aus dem Vektor der Drehwinkelgeschwindigkeit ${}^{\mathcal{O}}_{\hat{\mathcal{O}}}\boldsymbol{\omega}$ und der Rotationsmatrix ${}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H}$, vgl. (2.19). Die Erweiterung der translatorischen Geschwindigkeit und Winkelgeschwindigkeit (4.10a) berechnet sich mit (2.11) in der Form

$$ {}^{\mathcal{T}}_{\hat{\mathcal{O}}}\dot{\mathbf{y}} = \begin{bmatrix} {}^{\mathcal{T}}_{\hat{\mathcal{O}}}\dot{\mathbf{d}} \\ {}^{\mathcal{T}}_{\hat{\mathcal{O}}}\boldsymbol{\omega} \end{bmatrix} = \begin{bmatrix} {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\dot{\mathbf{d}} + {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\dot{\mathbf{R}}\, {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{d} + f_{\text{R}}\left({}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H}\right) {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{d}} \\ {}^{\mathcal{O}}_{\hat{\mathcal{O}}}\boldsymbol{\omega} + f_{\text{R}}\left({}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H}\right) {}^{\mathcal{T}}_{\mathcal{O}}\boldsymbol{\omega} \end{bmatrix} \ . \tag{4.17} $$

4.2.4 Trajektorienfolgeregelung

Der Ausgang des Systems ${}^{\mathcal{T}}_{\mathcal{O}}\mathbf{H}(\mathbf{q})$ aus (4.9) soll einer vorgegebenen Trajektorie folgen und dabei auf die Trajektorie stabilisiert werden. Dafür wird zunächst ein PD-Regler verwendet und dessen Stabilität gezeigt. Danach wird dieser auf einen PID-Regler erweitert und die Daten der Sensorfusion in das Regelkonzept integriert.

Für die Stabilisierung der Orientierung wird eine geeignete Darstellung des Orientierungsfehlers benötigt. Nach den Überlegungen von [18, 34] wird dafür der Vektorteil der Einheitsquaternion verwendet. Eine Einheitsquaternion $\boldsymbol{\rho}^{\text{T}} = \begin{bmatrix} \eta & \boldsymbol{\epsilon}^{\text{T}} \end{bmatrix}$ besteht aus einem Skalarteil $\eta \in \mathbb{R}$ und einem Vektorteil $\boldsymbol{\epsilon} \in \mathbb{R}^3$. Der Vektorteil der Einheitsquaternion wird symbolisch mit der Funktion $f_{\epsilon}(\mathbf{R})$ aus einer Rotationsmatrix $\mathbf{R}$ berechnet. Für die Berechnung der Quaternion wird auf die Arbeit von Dam et al. [29] verwiesen.

Die vorgegebene Trajektorie beschreibt die Bewegung des Werkzeugkoordinatensystems $\mathcal{T}$ in Bezug auf das Objektkoordinatensystems $\mathcal{O}$ und muss daher zweifach stetig differenzierbar sein. Die Trajektorie besteht aus der vorgegebenen Position ${}^{\mathcal{T}}_{\mathcal{O}}\mathbf{d}_{\text{d}}$, Geschwindigkeit ${}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{d}}_{\text{d}}$ und Beschleunigung ${}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{d}}_{\text{d}}$ sowie der Rotationsmatrix ${}^{\mathcal{T}}_{\mathcal{O}}\mathbf{R}_{\text{d}}$, dem Vektor der Drehwinkelgeschwindigkeit ${}^{\mathcal{T}}_{\mathcal{O}}\boldsymbol{\omega}_{\text{d}}$ und der Drehwinkelbeschleunigung ${}^{\mathcal{T}}_{\mathcal{O}}\dot{\boldsymbol{\omega}}_{\text{d}}$. Das PD-Regelgesetz für die Zustandsrückführung (4.12) lautet

$$ \mathbf{u} = \begin{bmatrix} {}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{d}}_{\text{d}} - \mathbf{K}_{\text{D,t}}\,\dot{\mathbf{e}}_{\text{t}} - \mathbf{K}_{\text{P,t}}\,\mathbf{e}_{\text{t}} \\ {}^{\mathcal{T}}_{\mathcal{O}}\dot{\boldsymbol{\omega}}_{\text{d}} - \mathbf{K}_{\text{D,r}}\,\mathbf{e}_{\omega} - \mathbf{K}_{\text{P,r}}\,\mathbf{e}_{\text{o}} \end{bmatrix} , \tag{4.18} $$

mit den Fehlertermen für die Translation

$$\ddot{\mathbf{e}}_\mathrm{t} = {}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{d}} - {}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{d}}_\mathrm{d} \tag{4.19a}$$

$$\dot{\mathbf{e}}_\mathrm{t} = {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{d}} - {}^{\mathcal{T}}_{\mathcal{O}}\dot{\mathbf{d}}_\mathrm{d} \tag{4.19b}$$

$$\mathbf{e}_\mathrm{t} = {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{d} - {}^{\mathcal{T}}_{\mathcal{O}}\mathbf{d}_\mathrm{d} \tag{4.19c}$$

und der Rotation

$$\dot{\mathbf{e}}_\omega = {}^{\mathcal{T}}_{\mathcal{O}}\dot{\boldsymbol{\omega}} - {}^{\mathcal{T}}_{\mathcal{O}}\dot{\boldsymbol{\omega}}_\mathrm{d} \tag{4.20a}$$

$$\mathbf{e}_\omega = {}^{\mathcal{T}}_{\mathcal{O}}\boldsymbol{\omega} - {}^{\mathcal{T}}_{\mathcal{O}}\boldsymbol{\omega}_\mathrm{d} \tag{4.20b}$$

$$\mathbf{e}_\mathrm{o} = f_\epsilon\left({}^{\mathcal{T}}_{\mathcal{O}}\mathbf{R}\left({}^{\mathcal{T}}_{\mathcal{O}}\mathbf{R}_\mathrm{d}\right)^\mathrm{T}\right) . \tag{4.20c}$$

Durch Einsetzen des Regelgesetzes (4.18) in (4.13) ergibt sich die Fehlerdynamik, aufgeteilt auf Translation und Rotation, zu

$$\mathbf{0} = \ddot{\mathbf{e}}_\mathrm{t} + \mathbf{K}_{\mathrm{D,t}}\dot{\mathbf{e}}_\mathrm{t} + \mathbf{K}_{\mathrm{P,t}}\mathbf{e}_\mathrm{t} \tag{4.21a}$$

$$\mathbf{0} = \dot{\mathbf{e}}_\omega + \mathbf{K}_{\mathrm{D,r}}\mathbf{e}_\omega + \mathbf{K}_{\mathrm{P,r}}\mathbf{e}_\mathrm{o} . \tag{4.21b}$$

Die asymptotische Stabilität der translatorischen Fehlerdynamik (4.21a) kann gezeigt werden, indem die Matrizen in $\mathbf{K}_{\mathrm{D,t}}$ und $\mathbf{K}_{\mathrm{P,t}}$ als Diagonalmatrizen mit den Einträgen $k^i_{\mathrm{D,t}}$ und $k^i_{\mathrm{P,t}}$, $i = 1, 2, 3$ gewählt werden. Dadurch kommt es zu einer Entkopplung der einzelnen Fehlerterme $(\mathbf{e}_\mathrm{t})_j$ und die Gleichung (4.21a) kann als drei charakteristische Gleichungen in der Form

$$(\ddot{\mathbf{e}}_\mathrm{t})_j + k^i_{\mathrm{D,t}}(\dot{\mathbf{e}}_\mathrm{t})_j + k^i_{\mathrm{P,t}}(\mathbf{e}_\mathrm{t})_j = 0\,, \qquad (i,j) = \{1,x\},\{2,y\},\{3,z\} \tag{4.22}$$

dargestellt werden. Durch eine negativ reelle Polvorgabe kann das charakteristische Polynom (4.22) als Hurwitz-Polynom vorgegeben werden, wodurch die asymptotische Stabilität gegeben ist. Für den PD-Regler ist das charakteristische Polynom genau dann ein Hurwitz-Polynom wenn die Koeffizienten $k^i_{\mathrm{P,t}}$ und $k^i_{\mathrm{D,t}}$ positiv sind.

Die asymptotische Stabilität von (4.21b) wird aufgrund der Nichtlinearität der Fehlerterme (4.20) in [34] über die direkte Methode von Lyapunov gezeigt. Für eine asymptotische Stabilität müssen die Matrizen $\mathbf{K}_{\mathrm{D,r}}$ und $\mathbf{K}_{\mathrm{P,r}}$ positiv definit sein.

Der Roboter wird in der Simulation mit einem Modellfehler simuliert, siehe Abschnitt 2.2.4. Durch den Modellfehler entsteht ein stationärer Regelfehler, welcher durch einen reinen PD-Regler nicht kompensiert werden kann. Um den stationären Regelfehler auszuregeln, wird der PD-Regler um einen Integratoranteil (4.18) auf einen PID-Regler gemäß

$$\mathbf{u} = \begin{bmatrix}\mathbf{u}_\mathrm{t}\\ \mathbf{u}_\mathrm{r}\end{bmatrix} = \begin{bmatrix} {}^{\mathcal{T}}_{\mathcal{O}}\ddot{\mathbf{d}}_\mathrm{d} - \mathbf{K}_{\mathrm{D,t}}\dot{\mathbf{e}}_\mathrm{t} - \mathbf{K}_{\mathrm{P,t}}\mathbf{e}_\mathrm{t} - \mathbf{K}_{\mathrm{I,t}}\int_0^t \mathbf{e}_\mathrm{t}\,\mathrm{d}\tau \\ {}^{\mathcal{T}}_{\mathcal{O}}\dot{\boldsymbol{\omega}}_\mathrm{d} - \mathbf{K}_{\mathrm{D,r}}\mathbf{e}_\omega - \mathbf{K}_{\mathrm{P,r}}\mathbf{e}_\mathrm{o} - \mathbf{K}_{\mathrm{I,r}}\int_0^t \mathbf{e}_\mathrm{o}\,\mathrm{d}\tau \end{bmatrix} \tag{4.23}$$

erweitert. Für den translatorischen Teil folgt der Beweis der asymptotische Stabilität analog zum PD-Regler. Für das nichtlineare Fehlersystem der Orientierung (4.23) liegt kein systematischer Stabilitätsbeweis vor. Die Stabilität des geschlossenen Kreises wird mittels Simulationsstudien überprüft und bestätigt.

Durch Ersetzen des Ausgangs des nominellen Systems mit dem Ausgang der Sensorfusion (4.15) und (4.17) ergibt sich der erweiterte PID-Regler zu

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_\mathrm{t} \\ \mathbf{u}_\mathrm{r} \end{bmatrix} = \begin{bmatrix} {}_{\mathcal{O}}^{\mathcal{T}}\ddot{\mathbf{d}}_\mathrm{d} - \mathbf{K}_{\mathrm{D,t}}\,\dot{\hat{\mathbf{e}}}_\mathrm{t} - \mathbf{K}_{\mathrm{P,t}}\,\hat{\mathbf{e}}_\mathrm{t} - \mathbf{K}_{\mathrm{I,t}} \int_0^t \hat{\mathbf{e}}_\mathrm{t}\,\mathrm{d}\tau \\ {}_{\mathcal{O}}^{\mathcal{T}}\dot{\boldsymbol{\omega}}_\mathrm{d} - \mathbf{K}_{\mathrm{D,r}}\,\hat{\mathbf{e}}_\omega - \mathbf{K}_{\mathrm{P,r}}\,\hat{\mathbf{e}}_\mathrm{o} - \mathbf{K}_{\mathrm{I,r}} \int_0^t \hat{\mathbf{e}}_\mathrm{o}\,\mathrm{d}\tau \end{bmatrix} \tag{4.24}$$

mit den Fehlertermen für die Translation

$$\dot{\hat{\mathbf{e}}}_\mathrm{t} = {}_{\hat{\mathcal{O}}}^{\mathcal{T}}\dot{\mathbf{d}} - {}_{\mathcal{O}}^{\mathcal{T}}\dot{\mathbf{d}}_\mathrm{d} \tag{4.25a}$$

$$\hat{\mathbf{e}}_\mathrm{t} = {}_{\hat{\mathcal{O}}}^{\mathcal{T}}\mathbf{d} - {}_{\mathcal{O}}^{\mathcal{T}}\mathbf{d}_\mathrm{d} \tag{4.25b}$$

und die Rotation

$$\hat{\mathbf{e}}_\omega = {}_{\hat{\mathcal{O}}}^{\mathcal{T}}\boldsymbol{\omega} - {}_{\mathcal{O}}^{\mathcal{T}}\boldsymbol{\omega}_\mathrm{d} \tag{4.26a}$$

$$\hat{\mathbf{e}}_\mathrm{o} = f_\epsilon\left({}_{\hat{\mathcal{O}}}^{\mathcal{T}}\mathbf{R}\left({}_{\mathcal{O}}^{\mathcal{T}}\mathbf{R}_\mathrm{d}\right)^\mathrm{T}\right) \tag{4.26b}$$

Für die Implementierung des Integrators wird für die Diskretisierung die Euler-Vorwärts-Methode in der Form

$$x_{k+1} = x_k + T_\mathrm{R} u_k \tag{4.27a}$$

$$y_k = x_k \tag{4.27b}$$

verwendet.

4.2.5 Nullraumregelung

In diesem Abschnitt wird auf die Regelung der Nulldynamik des KUKA LBR iiwa 14 R820 Industrieroboters eingegangen. Der Roboter hat sieben Freiheitsgrade, wovon sechs durch die vorgegebene Trajektorie bestimmt werden. Der verbleibende Freiheitsgrad des redundanten Robotersystems wird durch $\boldsymbol{\tau}_\mathrm{N}$ in (4.11) geregelt. Der Regler kann verwendet werden, um verschiedene Zusatzziele zu erreichen, siehe [18]. Mit der Zustandsrückführung

$$\boldsymbol{\tau}_\mathrm{N} = \left(\mathbf{I} - {}_{\mathcal{O}}^{\mathcal{T}}\mathbf{J}^\dagger(\mathbf{q})\,{}_{\mathcal{O}}^{\mathcal{T}}\mathbf{J}(\mathbf{q})\right)\mathbf{u}_\mathrm{N} \tag{4.28}$$

wird der neue Eingang $\mathbf{u}_\mathrm{N}$ in den Nullraum projiziert. In dieser Arbeit ist dessen Ziel die Gelenke im Nullraum möglichst nahe zum Nullpunkt zu regeln und zu stabilisieren. Dafür wird ein PD-Regler gemäß

$$\mathbf{u}_\mathrm{N} = -\mathbf{K}_{\mathrm{D,N}}\,\dot{\mathbf{q}} - \mathbf{K}_{\mathrm{P,N}}\,\mathbf{q} \tag{4.29}$$

verwendet. Durch die Wahl von $\mathbf{K}_{\mathrm{D,N}}$ und $\mathbf{K}_{\mathrm{P,N}}$ zu positiv definiten Matrizen folgt die asymptotische Stabilität des Nullraumreglers, siehe [31].

5 Simulations- und Parameterstudie

In diesem Kapitel werden die Ergebnisse der Parameterstudie präsentiert. Dafür wird in Abschnitt 5.1 der Aufbau der Simulation, die Einstellungen, Parameter und die verwendeten Variablen beschrieben. In Abschnitt 5.2 und Abschnitt 5.3 werden die Ergebnisse der Simulationen ohne bzw. mit Posenbestimmung verglichen. Abschnitt 5.4 beschäftigt sich mit dem Regelfehler und es werden die Ergebnisse unter Verwendung des PD-Reglers und des PID-Reglers aus Abschnitt 4.2.4 vorgestellt. Die Genauigkeit der Posenbestimmung wird in Abschnitt 5.5 untersucht. In Abschnitt 5.6 wird die Laufzeit der Posenbestimmung in Abhängigkeit der Anzahl der verwendeten Messpunkte verglichen. Abschließend werden alle Ergebnisse in Abschnitt 5.7 diskutiert.

5.1 Aufbau der Simulation

Im Folgenden wird in Abschnitt 5.1.1 der Aufbau des Regelkreises und die Interaktion der einzelnen Bestandteile der Simulation miteinander beschrieben. Danach werden in Abschnitt 5.1.2 die verwendeten Einstellungen und Parameter erläutert. In Abschnitt 5.1.3 werden die für die folgenden Abschnitte benötigten Variablen eingeführt.

5.1.1 Regelkreis

In Abbildung 5.1 ist der Regelkreis für die Simulation vereinfacht dargestellt. Von links beginnend wird die Trajektorie in der `Trajektorienplanung` (4.7) erzeugt und an den Regler weitergegeben. Der `Regler` (4.12), (4.28) mit (4.18) bzw. (4.24) berechnet die benötigten Drehmomente $\boldsymbol{\tau}$ aus der Differenz zwischen der vorgegebenen Trajektorie und der geschätzten Pose aus der Sensorfusion. Der Block `simulierter Roboter` (2.28), (2.29) berechnet die Bewegung des Roboters aus den Drehmomenten mithilfe der Dynamik des realen Roboters. Der Block hat zwei getrennte Ausgänge, nämlich die realen Posen $\tilde{\mathcal{E}}$ und $\tilde{\mathcal{O}}$ sowie die nominelle Pose $\mathcal{E}$ und $\mathcal{O}$. Der `simulierte Sensor` (Abschnitt 2.3.3)

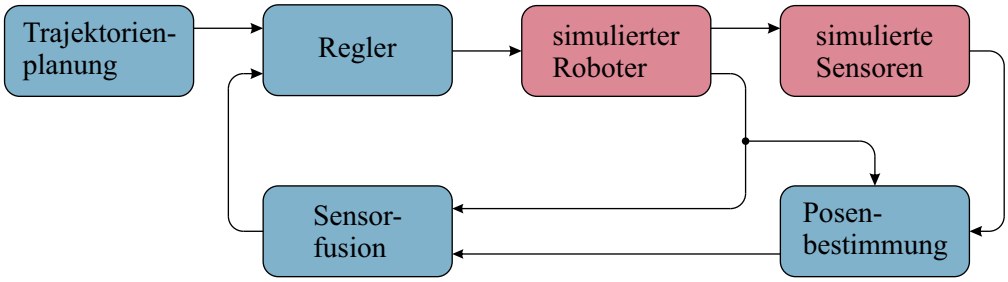


Abbildung 5.1: Vereinfachte Darstellung des Regelkreises.

berechnet die Messpunkte ${}_{\mathcal{S}}P_{\mathrm{S}}$ aus der realen Pose des Objektes $\tilde{\mathcal{O}}$ relativ zum Sensor. Die `Posenbestimmung` (Abschnitt 3.2.5) berechnet mithilfe des ApPzE-ICP-Algorithmus aus den Messpunkten ${}_{\mathcal{S}}P_{\mathrm{S}}$ und der nominellen Pose des Objektes $\mathcal{O}$ die geschätzte Pose $\hat{\mathcal{O}}_{\mathrm{icp}}$. Um einen stetigen Verlauf für die Regelung zu erhalten, wird diese gefiltert und die gefilterte Schätzung der Pose $\hat{\mathcal{O}}$ bezeichnet. Die `Sensorfusion` (4.14), (4.17) vereint die nominelle Pose des Industrieroboters mit der gefilterten Schätzung aus der Posenbestimmung. Diese Daten bilden die Rückführung für den Regler.

5.1.2 Parameter und Einstellungen

Im Folgenden werden die Einstellungen und die Parameter für die folgenden Simulationen eingeführt. Die entsprechenden Werte sind in Tabelle 5.1 zusammengefasst. Der translatorische Modellfehler $\tilde{\mathbf{d}}_{\mathrm{y}}$ wird für jedes Glied des Industrieroboters gleich gewählt. Der rotatorische Modellfehler $\tilde{\mathbf{q}}$ wird bei den letzten vier Gelenken angewendet. In den ersten drei Gliedern würde ein rotatorischer Modellfehler zu einer sehr großen translatorischen Verschiebungen des Endeffektors führen, weshalb dieser Modellfehler für diese Gelenke nicht verwendet wird. Der Modellfehler wird so gewählt, dass ein translatorischer Fehler von mehreren Millimetern und ein Fehler in der Orientierung von mehreren Grad entsteht. Die Werte für das Messrauschen des 2D-Lasertriangulationssensors werden entsprechend dem Datenblatt [22] bzw. Abschnitt 2.3.4 gewählt. Die Pose der Basis des Industrieroboters Kuka LBR iiwa 14 R820 $\mathcal{B}$, die Pose des Werkzeugs $\mathcal{T}$ und die Pose des 2D-Lasertriangulationssensors Micro-Epsilon scanCONTROL 2600-100 $\mathcal{S}$ sind in Kapitel A.1 als homogene Transformation in Bezug auf das Weltkoordinatensystems $\mathcal{W}$ notiert. Für den PID-Regler werden alle Pole des geschlossenen Regelkreises auf -30 gesetzt. Diese Pole sind für einen realen Aufbau auf einem Kuka LBR iiwa 14 R820 geeignet, wie bereits abgeschlossene Projekte am Institut für Automatisierungs- und Regelungstechnik [35] zeigen. Die Parameter der Posenbestimmung werden analog zur stationären Simulation gewählt, siehe Tabelle 3.3. Die Simulation des Roboters erfolgt als zeitkontinuierliches Modell, der Rest des Regelkreises wird zeitdiskret simuliert. Die Abtastzeiten werden wie folgt gewählt:

- T_{icp} entspricht der Abtastzeit der Posenbestimmung und des 2D-Lasertriangulationssensors und wird entsprechend der Laufzeit des ICP-Algorithmus gewählt, siehe Abschnitt 5.6.
- T_{R} entspricht der Abtastzeit des Reglers, der Trajektorienplanung, des Filters für die Posenbestimmung und der Sensorfusion und wird analog zum experimentellen Aufbau der Arbeit [35] gewählt.

Die Simulation erfolgt unter der Verwendung von Matlab/Simulink. Der `Simulation Mode` wird dabei auf `Accelerator` eingestellt um eine möglichst kurze Simulationszeit zu erhalten. Für die Simulation wird ein Laptop mit folgenden Eigenschaften verwendet:

- Betriebssystem: Windows 10 Build 19041
- Prozessor: Intel Core i7-4700MQ @ 2,4 GHz

- Arbeitsspeicher: 16 GB DDR3 @ 1600 MHz
- Software: MATLAB/SIMULINK 2018b

Tabelle 5.1: Parameter und Einstellungen für die Simulation.

Beschreibung	Gleichung	Symbol	Einheit	Wert
Modell-fehler	(2.26a)	$\tilde{\mathbf{d}}_\mathrm{y}$	mm	$\begin{bmatrix}2 & 2 & 2 & 2 & 2 & 2 & 2\end{bmatrix}^\mathrm{T}$
	(2.26b)	$\tilde{\mathbf{q}}$	°	$\begin{bmatrix}0 & 0 & 0 & 0{,}5 & 1 & 1 & 1\end{bmatrix}^\mathrm{T}$
Mess-rauschen	(2.43)	μ_E	m	0
	(2.43)	σ_E	µm	65
PID-Regler	(4.24)	$\mathrm{diag}(\mathbf{K}_{\mathrm{D,t}})$	1/s	$\begin{bmatrix}90 & 90 & 90\end{bmatrix}$
	(4.24)	$\mathrm{diag}(\mathbf{K}_{\mathrm{D,r}})$	1/s	$\begin{bmatrix}90 & 90 & 90\end{bmatrix}$
	(4.24)	$\mathrm{diag}(\mathbf{K}_{\mathrm{P,t}})$	$1/\mathrm{s}^2$	$2{,}7 \cdot 10^3\begin{bmatrix}1 & 1 & 1\end{bmatrix}$
	(4.24)	$\mathrm{diag}(\mathbf{K}_{\mathrm{P,r}})$	$1/\mathrm{s}^2$	$2{,}7 \cdot 10^3\begin{bmatrix}1 & 1 & 1\end{bmatrix}$
	(4.24)	$\mathrm{diag}(\mathbf{K}_{\mathrm{I,t}})$	$1/\mathrm{s}^3$	$27 \cdot 10^3\begin{bmatrix}1 & 1 & 1\end{bmatrix}$
	(4.24)	$\mathrm{diag}(\mathbf{K}_{\mathrm{I,r}})$	$1/\mathrm{s}^3$	$27 \cdot 10^3\begin{bmatrix}1 & 1 & 1\end{bmatrix}$
Abtast-zeiten	-	T_icp	ms	50
	(3.26)	T_R	µs	125
Pfad & Trajektorie	(4.3)	l_p	mm	180
	(4.3)	v_d	mm/s	7
	(4.3)	T_init	s	1
Nullraum-regler	(4.29)	$\mathrm{diag}(\mathbf{K}_{\mathrm{D,N}})$	N m	$\begin{bmatrix}10 & 10 & 10 & 10 & 10 & 10 & 10\end{bmatrix}$
	(4.29)	$\mathrm{diag}(\mathbf{K}_{\mathrm{P,N}})$	N m s	$\begin{bmatrix}25 & 25 & 25 & 25 & 25 & 25 & 25\end{bmatrix}$
PD-Regler	(4.18)	$\mathrm{diag}(\mathbf{K}_{\mathrm{D,t}})$	1/s	$\begin{bmatrix}60 & 60 & 60\end{bmatrix}$
	(4.18)	$\mathrm{diag}(\mathbf{K}_{\mathrm{D,r}})$	1/s	$\begin{bmatrix}60 & 60 & 60\end{bmatrix}$
	(4.18)	$\mathrm{diag}(\mathbf{K}_{\mathrm{P,t}})$	$1/\mathrm{s}^2$	$1 \cdot 10^2\begin{bmatrix}9 & 9 & 9\end{bmatrix}$
	(4.18)	$\mathrm{diag}(\mathbf{K}_{\mathrm{P,r}})$	$1/\mathrm{s}^2$	$1 \cdot 10^2\begin{bmatrix}9 & 9 & 9\end{bmatrix}$

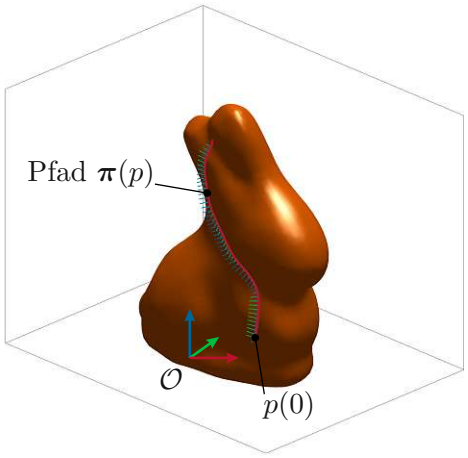


Abbildung 5.2: Darstellung des Pfades $\boldsymbol{\pi}(p)$ am Objekt.

Der Pfad $\boldsymbol{\pi}(p)$ auf dem Objekt ist in Abbildung 5.2 dargestellt. Der Pfad wird beginnend von unten nach oben mit dem Werkzeug abgefahren. Wie in Abschnitt 4.1 beschrieben, wird jedem Punkt auf der Trajektorie ein Kontaktkoordinatensystem $\mathcal{K}$ zugeordnet, siehe (4.1). Die Eigenschaften des Pfades und der Trajektorie sind in Tabelle 5.1 zusammengefasst.

5.1.3 Variablen

In diesem Abschnitt werden jene Variablen erläutert, welche in den Zeitverläufen, Diagrammen und Ergebnissen der Simulations- und Parameterstudie dargestellt sind. Dafür werden zuerst die benötigten Posen und anschließend die betrachteten Fehlermetriken eingeführt.

Die vorgegebene Trajektorie wird als homogene Transformation ${}^{\mathcal{K}}_{\mathcal{O}}\mathbf{H}_{\mathrm{d}}$ am Objekt mit der nominellen Pose $\mathcal{O}$ vorgegeben. In der Simulation soll die Trajektorie mithilfe der Schätzung der realen Pose des Objektes $\tilde{\mathcal{O}}$ durch den Roboter abgefahren werden. Der entsprechende Kontaktpunkt wird mit $\tilde{\mathcal{K}}$ bezeichnet. Das Ziel des Ablaufes ist es, den Abstand zum Werkzeugkoordinatensystem $\mathcal{T}$ möglichst gering zu halten. Durch die Schätzung mithilfe der Posenbestimmung ${}^{\mathcal{T}}_{\hat{\mathcal{O}}}\mathbf{H}$ ergibt sich die geschätzte Pose des Objektes $\hat{\mathcal{O}}$, das zugehörige Koordinatensystem im Kontaktpunkt wird mit $\hat{\mathcal{K}}$ bezeichnet. Der Zusammenhang zwischen den realen Posen und den nominellen Posen ist in Abbildung 3.2 dargestellt.

Für die Beschreibung der Genauigkeit wird ein skalares Maß des Abstandes bzw. die Änderungsrate des Abstandes zwischen verschiedenen Posen benötigt. Die Metrik für die Genauigkeit wird auf einen translatorischen und rotatorischen Anteil aufgeteilt. Die allgemeinen Fehlermetriken zwischen den Koordinatensystemen $\mathcal{X}$ und $\mathcal{Y}$ lauten

$$f_{\mathrm{K}}(\mathcal{X},\mathcal{Y}) = \begin{bmatrix} f_{\mathrm{K,d}}(\mathcal{X},\mathcal{Y}) & f_{\mathrm{K,r}}(\mathcal{X},\mathcal{Y}) \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^2 \tag{5.1a}$$

$$\text{mit} \quad f_{\mathrm{K,d}}(\mathcal{X},\mathcal{Y}) = \left\| f_{\mathrm{d}}\left({}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{H}\right) \right\| \tag{5.1b}$$

$$f_{\mathrm{K,r}}(\mathcal{X},\mathcal{Y}) = f_{\mathrm{E,R}}\left({}^{\mathcal{Y}}_{\mathcal{X}}\mathbf{H}\right) . \tag{5.1c}$$

Die Funktion $f_{\mathrm{E,R}}(\cdot)$ berechnet die Gesamtrotation der homogenen Transformation gemäß (3.18). Für die Berechnung der Änderungsrate des Abstandes zwischen den Koordinatensystemen $\mathcal{X}$ und $\mathcal{Y}$ wird die Funktion

$$f_{\mathrm{K,v}}(\mathcal{X},\mathcal{Y}) = \left\| f_{\mathrm{d}}\left({}^{\mathcal{Y}}_{\mathcal{X}}\dot{\mathbf{H}}\right)\right\| \tag{5.2}$$

verwendet.

Das Endeffektorkoordinatensystem $\mathcal{E}$, das Objektkoordinatensystem $\mathcal{O}$ und das Kontaktkoordinatensystem $\mathcal{K}$ werden in dieser Arbeit in drei Varianten verwendet. Die Variante mit Tilde, d. h. $\tilde{\mathcal{E}}$, $\tilde{\mathcal{O}}$ und $\tilde{\mathcal{K}}$ entspricht der realen Pose dieser Koordinatensysteme im Raum. Die nominelle Variante $\mathcal{E}$, $\mathcal{O}$ und $\mathcal{K}$ bezeichnet die Pose, welche sich aufgrund der nominellen Parameter ergibt. Die Variante mit Dach, d. h. $\hat{\mathcal{E}}$, $\hat{\mathcal{O}}$ und $\hat{\mathcal{K}}$ entspricht der Pose, welche sich durch die Schätzung aus der Posenbestimmung ergibt.

Für die Diskussion der Ergebnisse aus der Simulations- und Parameterstudie werden folgenden Metriken eingeführt und betrachtet:

- $\begin{bmatrix} {}^{\mathcal{K}}_{\tilde{\mathcal{K}}}d & {}^{\mathcal{K}}_{\tilde{\mathcal{K}}}r \end{bmatrix}^{\mathrm{T}} = f_{\mathrm{K}}\left(\tilde{\mathcal{K}},\mathcal{K}\right)$ ist der Abstand zwischen der realen Pose des Kontaktpunktes $\tilde{\mathcal{K}}$ und der nominellen Pose des Kontaktpunktes $\mathcal{K}$ und entsteht durch den Modellfehler, siehe Abschnitt 2.2.4.
- $\begin{bmatrix} {}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}d & {}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r \end{bmatrix}^{\mathrm{T}} = f_{\mathrm{K}}\left(\tilde{\mathcal{K}},\hat{\mathcal{K}}\right)$ ist der Abstand zwischen der realen Pose des Kontaktpunktes $\tilde{\mathcal{K}}$ und der geschätzten Pose des Kontaktpunktes $\hat{\mathcal{K}}$ und wird als Fehler in der Posenbestimmung bezeichnet.
- $\begin{bmatrix} {}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}d_{\mathrm{icp}} & {}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{\mathrm{icp}} \end{bmatrix}^{\mathrm{T}} = f_{\mathrm{K}}\left(\tilde{\mathcal{K}},\hat{\mathcal{K}}_{\mathrm{icp}}\right)$ ist der Abstand zwischen der realen Pose des Kontaktpunktes $\tilde{\mathcal{K}}$ und der geschätzten Pose des Kontaktpunktes $\hat{\mathcal{K}}_{\mathrm{icp}}$ ohne die Filterung und wird als Fehler in der Posenbestimmung ohne Filterung bezeichnet. Dieser Fehler wird nicht im Regelkreis verwendet, ist aber von Interesse, um die Genauigkeit der Posenbestimmung zu beurteilen.
- $\begin{bmatrix} {}^{\mathcal{T}}_{\hat{\mathcal{K}}}d & {}^{\mathcal{T}}_{\hat{\mathcal{K}}}r \end{bmatrix}^{\mathrm{T}} = f_{\mathrm{K}}\left(\hat{\mathcal{K}},\mathcal{T}\right)$ ist der Abstand zwischen der geschätzten Pose des Kontaktpunktes $\hat{\mathcal{K}}$ und der Werkzeugpose $\mathcal{T}$. Dieser Fehler wird als Regelfehler bezeichnet. Dabei ist zu beachten, dass diese Fehlermetrik nur zweidimensional ist, siehe (5.1a), und nicht den sechsdimensionalen Regelfehler im Regler entspricht. Der so berechnete Regelfehler erlaubt eine bessere Bewertung der Performance des Reglers.
- $\begin{bmatrix} {}^{\mathcal{T}}_{\tilde{\mathcal{K}}}d & {}^{\mathcal{T}}_{\tilde{\mathcal{K}}}r \end{bmatrix}^{\mathrm{T}} = f_{\mathrm{K}}\left(\tilde{\mathcal{K}},\mathcal{T}\right)$ ist der Abstand zwischen der realen Pose des Kontaktpunktes $\tilde{\mathcal{K}}$ und der Werkzeugpose $\mathcal{T}$ und wird als Gesamtfehler bezeichnet. Der Gesamtfehler besteht aus dem Fehler in der Posenbestimmung sowie dem Regelfehler.
- ${}^{\mathcal{K}}_{\mathcal{O}}v_{\mathrm{d}} = f_{\mathrm{K,v}}(\mathcal{O},\mathcal{K})$ ist die Geschwindigkeit der vorgegebenen Trajektorie des Kontaktpunktes $\mathcal{K}$ in Bezug auf das Objektkoordinatensystem $\mathcal{O}$.
- ${}^{\mathcal{T}}_{\tilde{\mathcal{O}}}v = f_{\mathrm{K,v}}\left(\tilde{\mathcal{O}},\mathcal{T}\right)$ ist die Geschwindigkeit der Werkzeugpose $\mathcal{T}$ in Bezug auf die reale Pose des Objektes $\tilde{\mathcal{O}}$. Dies entspricht der Geschwindigkeit mit der das Werkzeug die Trajektorie auf dem Objekt mit der realen Pose $\tilde{\mathcal{O}}$ abfährt.

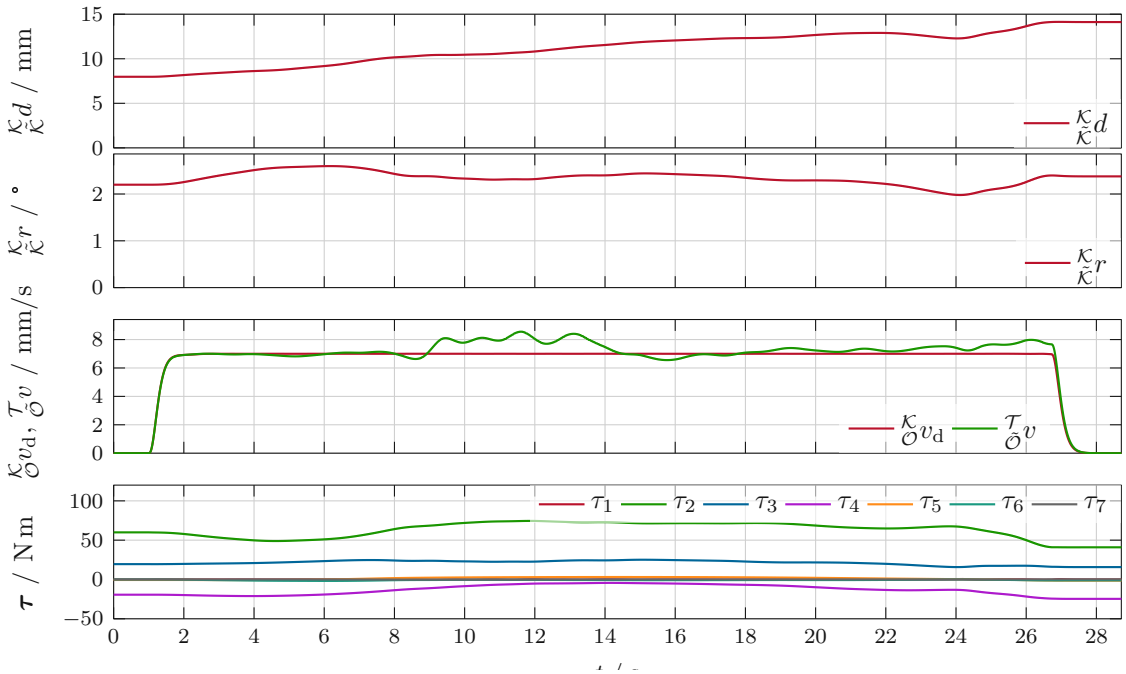


Abbildung 5.3: Ergebnisse der Simulation ohne Posenbestimmung mit dem Modellfehler, der Pfadgeschwindigkeit und den Drehmomenten.

- ${}^{\tilde{\mathcal{E}}}_{\mathcal{B}}v = f_{\mathrm{K,v}}\left(\mathcal{B}, \tilde{\mathcal{E}}\right)$ beschreibt die Geschwindigkeit des realen Endeffektorkoordinatensystems $\tilde{\mathcal{E}}$ in Bezug auf das Basiskoordinatensystem des Roboters $\mathcal{B}$.

5.2 Simulation ohne Posenbestimmung

In diesem Abschnitt wird das Ergebnis der Simulation mit deaktivierter Posenbestimmung präsentiert. Dafür wird das Ergebnis der Posenbestimmung ${}^{\mathcal{O}}_{\tilde{\mathcal{O}}}\mathbf{H}$ konstant auf die Einheitsmatrix gesetzt, d. h. ${}^{\mathcal{O}}_{\tilde{\mathcal{O}}}\mathbf{H} = \mathbf{E}$, wodurch es in der Sensorfusion zu keiner Veränderung der nominellen Posen $\mathcal{E}$ und $\mathcal{O}$ kommt. Für die Regelung wird der PID-Regler verwendet. Der Rest des Regelkreises ist unverändert. Der Industrieroboter wird mit der Dynamik des realen Roboters, $\tilde{\mathbf{d}}_{\mathrm{y}}$ und $\tilde{\mathbf{q}}$ laut Tabelle 5.1, simuliert. Die Regelung arbeitet hingegen mit den nominellen Parametern des Roboters

In Abbildung 5.3 sind in den oberen zwei Graphen die translatorische Fehlermetrik ${}^{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und die rotatorische Fehlermetrik ${}^{\mathcal{K}}_{\tilde{\mathcal{K}}}r$, welche zufolge des Modellfehler entstehen, dargestellt. Der in Abschnitt 2.2.4 eingeführte Modellfehler führt zu einer Differenz zwischen der nominellem und realen Pose des Kontaktpunktes $\mathcal{K}$ bzw. $\tilde{\mathcal{K}}$. Durch die Bewegung des Roboters ändert sich dieser Fehler zeitabhängig. Der Fehler bewegt sich im Bereich von 8 mm bis 14 mm translatorisch, sowie von 2° bis 2,6° rotatorisch. Im dritten Graph ist die vorgegebene Trajektoriengeschwindigkeit ${}^{\mathcal{K}}_{\mathcal{O}}v_{\mathrm{d}}$ sowie die reale Oberflächengeschwindigkeit des Werkzeugs ${}^{\mathcal{T}}_{\tilde{\mathcal{O}}}v$ dargestellt. Zu Beginn und am Ende der Simulation ist der Industrieroboter im Stillstand. Im Zeitraum dazwischen wird die Sollgeschwindigkeit ${}^{\mathcal{K}}_{\mathcal{O}}v_{\mathrm{d}}$ vorgegeben.

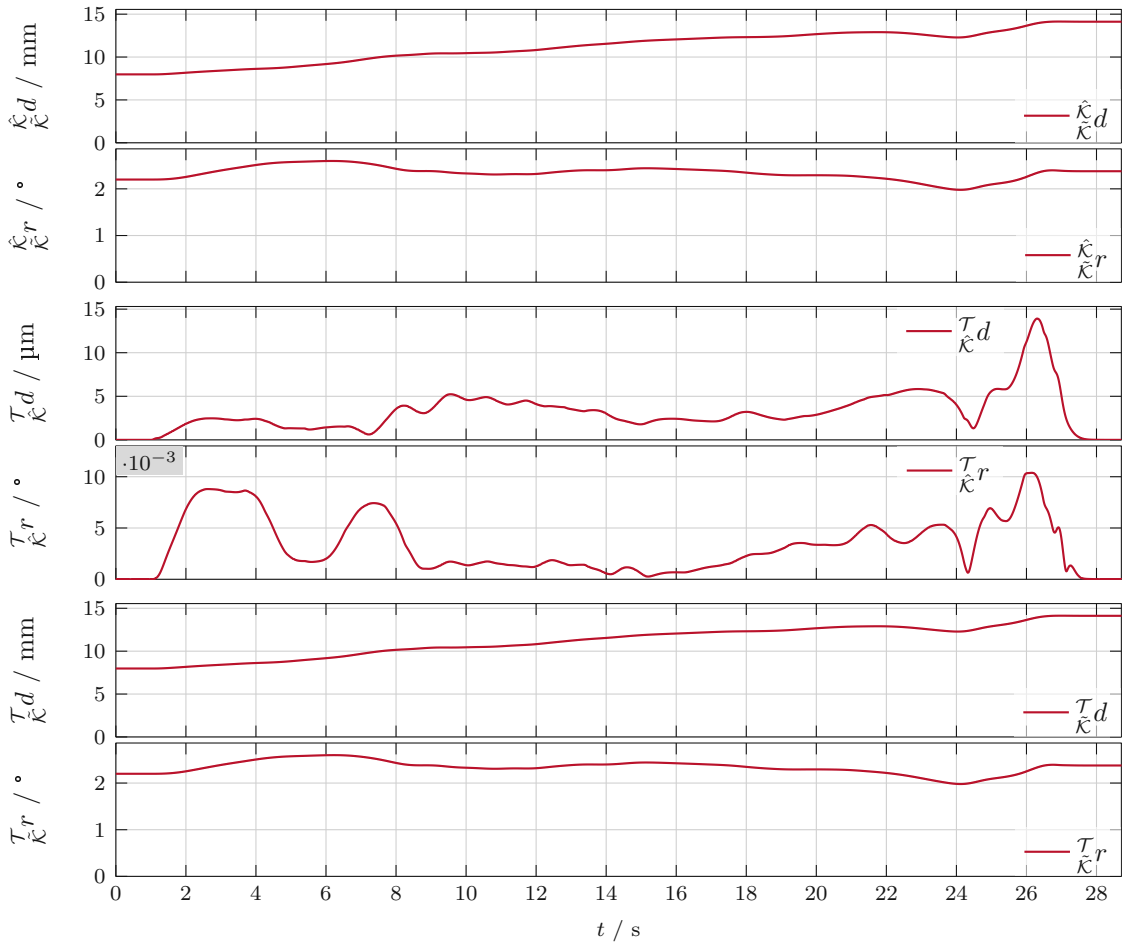


Abbildung 5.4: Ergebnisse der Simulation ohne Posenbestimmung mit dem Fehler in der Posenbestimmung, Regelfehler und Gesamtfehler.

Durch die Filterung des Pfadparameters $\bar{p}$ entsteht ein glatter Verlauf der Geschwindigkeit, siehe Abschnitt 4.1.2. Aufgrund der Abweichung zwischen realem und nominellem Modell weicht die erreichte Geschwindigkeit ${}^{\mathcal{T}}_{\tilde{\mathcal{O}}}v$ sichtbar von der vorgegebenen Geschwindigkeit ${}^{\mathcal{K}}_{\tilde{\mathcal{O}}}v_\mathrm{d}$ ab. Im letzten Graphen sind die sieben Motordrehmomente $\boldsymbol{\tau}$ dargestellt. Aufgrund des glatten und gleichmäßigen Verlaufs der Geschwindigkeit ${}^{\mathcal{T}}_{\tilde{\mathcal{O}}}v$ ergibt sich ein glatter Verlauf der Drehmomente $\boldsymbol{\tau}$.

In Abbildung 5.4 sind in den oberen zwei Graphen die Fehlermetriken ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}d$ und ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r$, welche den Fehler in der Posenbestimmung beschreiben, dargestellt. In dieser Simulation ist die Posenbestimmung deaktiviert, d. h. ${}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H} = \mathbf{E}$. Infolgedessen sind die Verläufe dieser Fehlermetriken ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}d$ und ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r$ die selben wie bei den Fehlermetriken zufolge des Modellfehlers ${}^{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und ${}^{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ von Abbildung 5.3. Der dritte und vierte Graph stellen jeweils die Fehlermetrik des Regelfehlers ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}d$ bzw. ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}r$ dar. Dieser ist im Vergleich zum Modellfehler ${}^{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ bzw. ${}^{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ mit maximal 14 µm und 10 Milligrad sehr klein. Die Fehlermetriken des Gesamtfehlers

${}^{\mathcal{T}}_{\tilde{\mathcal{K}}}d$ und ${}^{\mathcal{T}}_{\tilde{\mathcal{K}}}r$ werden in den letzten beiden Graphen in Abbildung 5.4 dargestellt. Diese Metrik entspricht dem Abstand zwischen der realen Pose des Kontaktpunktes $\tilde{\mathcal{K}}$ zur Werkzeugpose $\mathcal{T}$ und entspricht daher der Kombination von Modell- und Regelfehler. Das Ziel dieser Arbeit ist es, den Gesamtfehler zu minimieren um eine korrekte Bearbeitung des Objektes durch das Werkzeug entlang der vorgegebenen Trajektorie zu erhalten.

5.3 Simulation mit Posenbestimmung

In diesem Abschnitt wird das Ergebnis der Simulation mit aktivierter Posenbestimmung präsentiert, siehe Abschnitt 3.2.5. Die sonstigen Einstellungen sind analog zum vorherigen Experiment, siehe Abschnitt 5.2. Das Messrauschen des 2D-Lasertriangulationssensors ist aktiviert mit einer Standardabweichung von $\sigma_{\mathrm{E}} = 65\,\mu\mathrm{m}$.

Der Modellfehler entspricht der Differenz zwischen der realen Pose des Kontaktpunktes $\tilde{\mathcal{K}}$ und seiner nominellen Pose $\mathcal{K}$. Da diese Metrik unabhängig von der geschätzten Pose des Kontaktpunktes $\hat{\mathcal{K}}$ ist, ist diese unabhängig von der Posenbestimmung und der Regelung. Aus diesem Grund bleibt dieser Modellfehler unverändert zum vorherigen Experiment, siehe Abbildung 5.3.

In Abbildung 5.5 sind in den oberen zwei Graphen die Fehlermetriken der Posenbestimmung ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}d$ und ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r$ dargestellt. Die aktivierte Posenbestimmung führt zu einer Reduktion des maximalen Wertes der Fehlermetriken der Posenbestimmung. Der translatorische Fehler ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}d$ wird von 14 mm auf 0,6 mm reduziert, während der rotatorische Fehler ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r$ von 2,6° auf 1,5° fällt, vgl. Abbildung 5.4. Es ist auch zu beachten, dass der Fehler in der Posenbestimmung über die Zeit variiert. Im Zeitraum 1 (7 s bis 15 s) und Zeitraum 2 (24 s bis 27 s) ist der Wert der Fehlermetrik relativ groß gegenüber den restlichen Abschnitten der Simulation. Um diese Unterschiede aufzuklären, wird in Abschnitt 5.5 die Genauigkeit der Posenbestimmung untersucht. Dabei konnten mehrere Ursachen für den zeitveränderlichen Fehler identifiziert werden. Die Fehlermetrik des Regelfehlers ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}d$ und ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}r$ ist in den Graphen drei und vier dargestellt. Gegenüber der Simulation ohne Posenbestimmung ist der Wert der Fehlermetrik des Regelfehlers translatorisch von 14 µm auf 100 µm und rotatorisch von 10 Milligrad auf 0,3° gestiegen, vgl. Abbildung 5.4. Die Posenbestimmung bringt über die Sensorfusion eine Änderung in die Rückführung des Reglers ein, wodurch es kurzfristig zu einem erhöhten Regelfehler kommt bis dieser ausgeregelt wird. Dieser Effekt wiederholt sich mit der Abtastzeit der Posenbestimmung T_{icp} und wird durch den Filter der Posenbestimmung abgeflacht, siehe Abschnitt 3.3.2. Die Fehlermetrik des Gesamtfehlers ${}^{\mathcal{T}}_{\tilde{\mathcal{K}}}d$ und ${}^{\mathcal{T}}_{\tilde{\mathcal{K}}}r$ wird trotzdem hauptsächlich vom Fehler in der Posenbestimmung bestimmt, vgl. Graphen 5 und 6 in Abbildung 5.5. In Abbildung 5.6 ist im ersten Graphen zu erkennen, dass die Geschwindigkeit ${}^{\mathcal{T}}_{\mathcal{O}}v$ der vorgegebenen Trajektoriengeschwindigkeit ${}^{\mathcal{K}}_{\mathcal{O}}v_{\mathrm{d}}$ besser folgt als in der vorherigen Simulation ohne Posenbestimmung in Abschnitt 5.2. Die Spitzen in der Geschwindigkeit ${}^{\mathcal{T}}_{\mathcal{O}}v$ entstehen weil der Regler die Änderungen in der Pose, welche durch die Posenbestimmung entstehen, kompensiert. Dadurch kommt es auch zu Spitzen im Drehmoment $\boldsymbol{\tau}$, dargestellt in Graph 2 in Abbildung 5.6. Im Bereich von Zeitraum 1 kommt es zu hohen Änderungen des Fehlers in der Posenbestimmung, wodurch viel korrigiert werden muss, weshalb die Spitzen in den Drehmomentsignalen hoch sind. Auch gibt es am Anfang und am Ende in der Ruhezeit aufgrund von kleinen

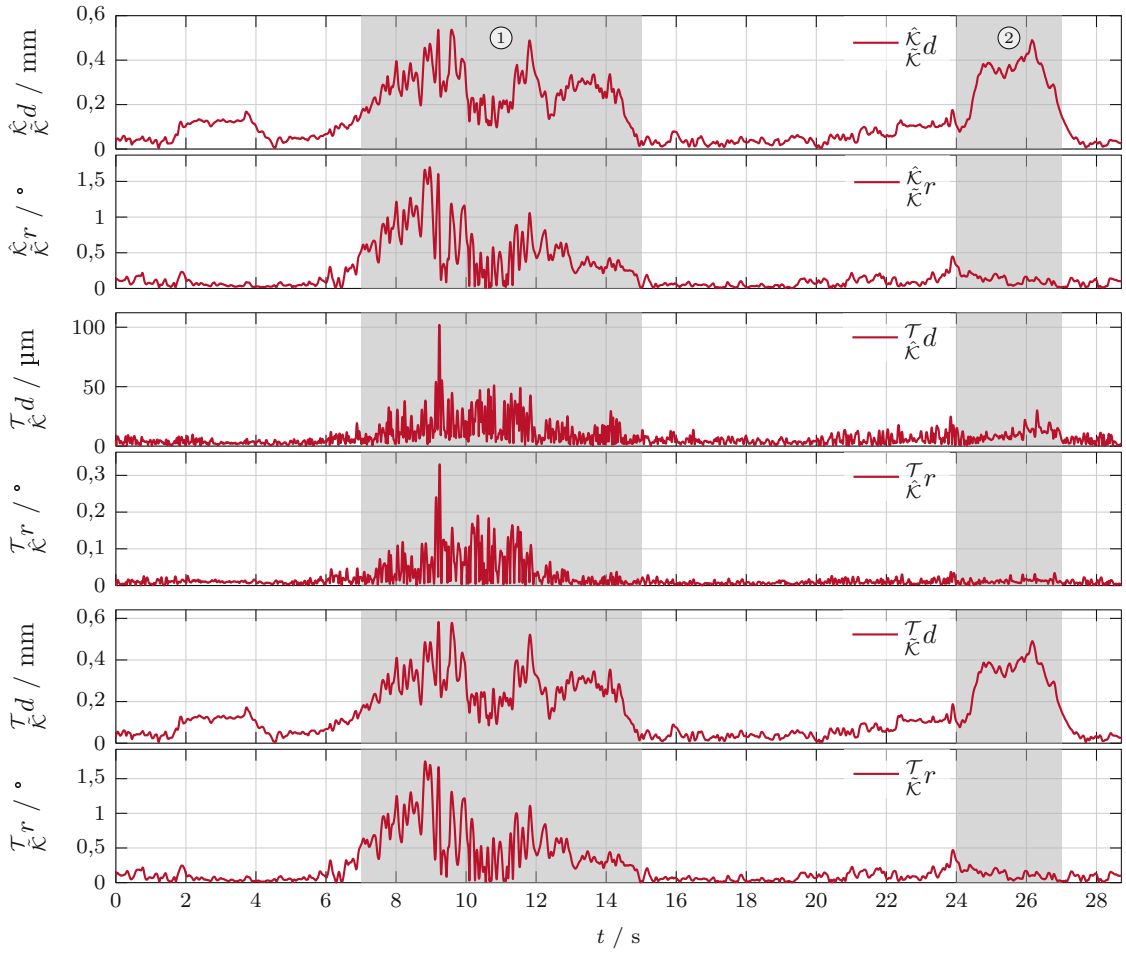


Abbildung 5.5: Ergebnisse der Simulation mit Posenbestimmung mit dem Fehler in der Posenbestimmung, Regelfehler und Gesamtfehler.

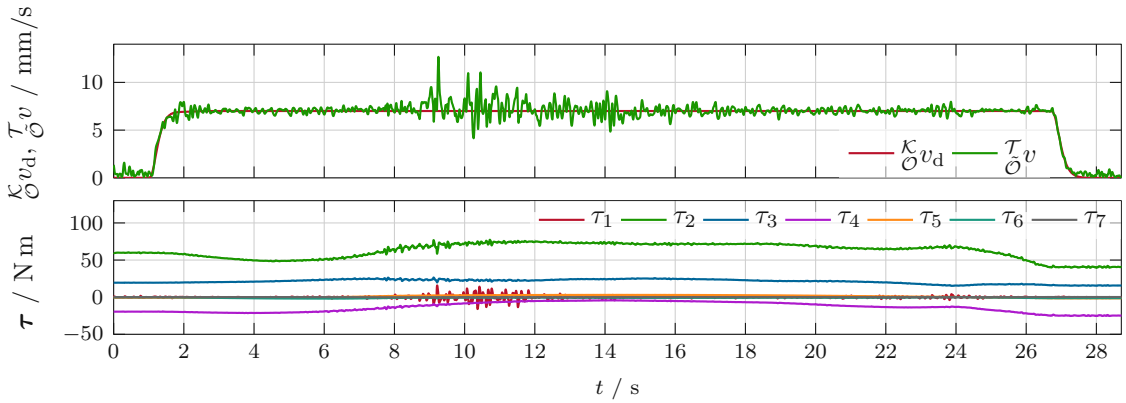


Abbildung 5.6: Ergebnisse der Simulation mit Posenbestimmung mit der Pfadgeschwindigkeit und den Drehmomenten.

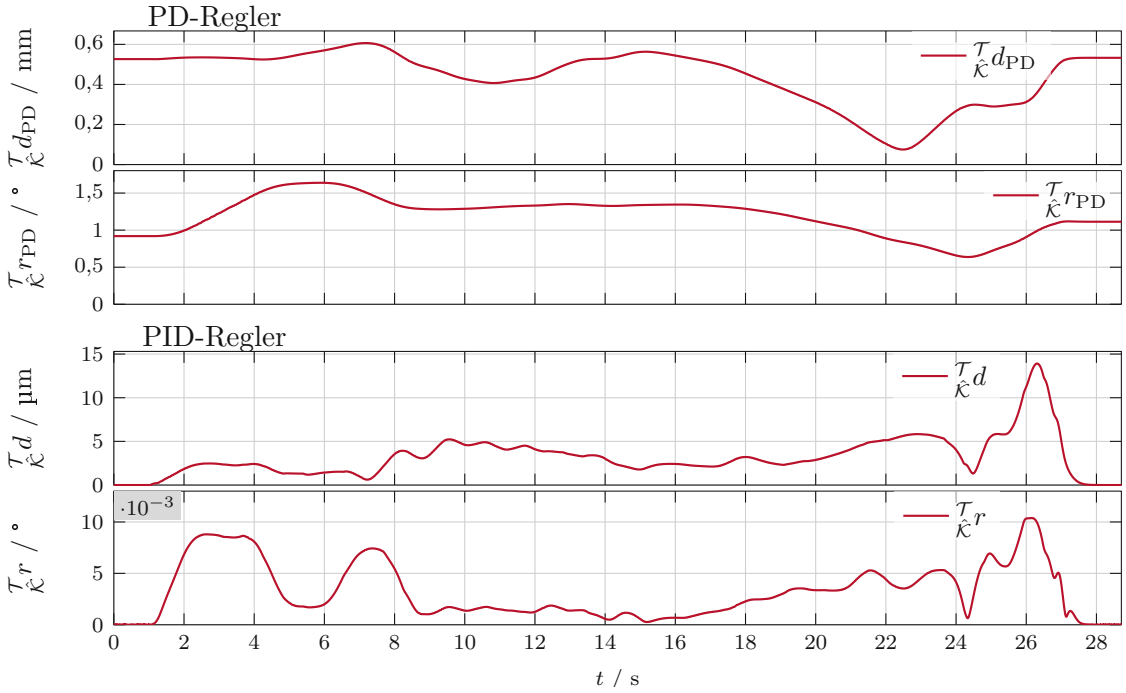


Abbildung 5.7: Vergleich des Regelfehlers von dem PD-Regler und dem PID-Regler.

Positionsänderungen eine geringe Geschwindigkeit ${}^{\mathcal{T}}_{\hat{\mathcal{O}}}v$ des Roboters, welche aufgrund des Rauschens der Posenbestimmung entsteht.

5.4 Untersuchung des Regelfehlers

In diesem Abschnitt werden die Regelfehler des PD-Reglers sowie des PID-Reglers untersucht und miteinander verglichen. Die Pole des geschlossenen Regelkreises werden beim PD-Regler auf die selben Pole wie beim PID-Reglers gelegt, siehe Tabelle 5.1.

Um in diesem Experiment den Einfluss der Posenbestimmung auf die Regelung zu minimieren, wird analog zu Abschnitt 5.2 die Posenbestimmung deaktiviert, d. h. ${}^{\mathcal{O}}_{\hat{\mathcal{O}}}\mathbf{H} = \mathbf{E}$. Das Ergebnis ist in Abbildung 5.7 dargestellt. Die oberen zwei Graphen stellen die Fehlermetrik des Regelfehlers des PD-Reglers ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}d_{\mathrm{PD}}$ und ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}r_{\mathrm{PD}}$ dar und die unteren zwei Graphen zeigen die Fehlermetrik des Regelfehlers des PID-Reglers ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}d_{\mathrm{PD}}$ und ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}r_{\mathrm{PD}}$.

Der Regelfehler des PD-Reglers hat dabei einen maximalen translatorischen Fehler ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}d_{\mathrm{PD}}$ von 0,6 mm und einen rotatorischen Fehler ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}r_{\mathrm{PD}}$ von 1,6°. Der Regelfehler wird dabei nie zur Gänze ausgeregelt, auch nicht im Bereich der Ruhezeit am Anfang und am Ende der Simulation. Dagegen hat der Regelfehler des PID-Reglers einen maximalen translatorischen Fehler ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}d$ von 14 µm und einen rotatorischen Fehler ${}^{\mathcal{T}}_{\hat{\mathcal{K}}}r$ von 10 Milligrad. In dem stationären Bereich, d. h. am Anfang und am Ende der Simulation, wird der Fehler zur Gänze ausgeregelt.

Durch den Modellfehler, Abschnitt 2.2.4, passen die Geometrie, Kinematik und indirekt auch die Dynamik zwischen den Modellen mit nominellen und realen Parametern nicht

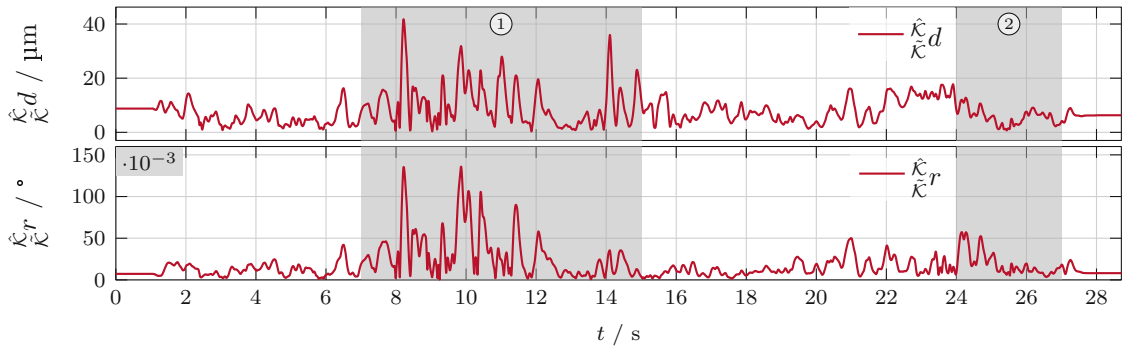


Abbildung 5.8: Fehlermetriken der Posenbestimmung, wenn Messrauschen und der Modellfehler deaktiviert sind.

zusammen. Durch die Unterschiede zum realen Modell, welches im simulierten Roboter verwendet wird, und dem nominellen Modell, welches im Regler in Verwendung ist, ist der PD-Regler nicht geeignet, um den Regelfehler entsprechend niedrig zu halten bzw. stationär zu unterdrücken. Aus diesem Grund ist der PID-Regler besser für die betrachtete Anwendung geeignet.

5.5 Untersuchung der Genauigkeit der Posenbestimmung

In diesem Abschnitt werden die Ergebnisse der Posenbestimmung genauer untersucht und die Zeitverläufe der Fehlermetriken der Posenbestimmung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ aus der Simulation diskutiert.

Wie in Abbildung 5.5 zu sehen ist, gibt es in dem Experiment mit aktivierter Posenbestimmung zwei Zeiträume bei der die Fehlermetriken der Posenbestimmung im Vergleich zur restlichen Simulation merklich größer sind, vgl. Abschnitt 5.3. Zeitraum 1 liegt zwischen 7 s und 15 s und Zeitraum 2 zwischen 24 s und 27 s.

5.5.1 Simulation ohne Messrauschen und ohne Modellfehler

In Abbildung 5.8 sind die Fehlermetriken der Posenbestimmung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ für eine Simulation mit deaktiviertem Messrauschen des 2D-Lasertriangulationssensors und ohne Modellfehler des Industrieroboters zu sehen. Der Positionsfehler ist im Vergleich zu Abbildung 5.5 sehr klein mit einem maximalen Fehler von 40 µm und 140 Milligrad, wodurch gezeigt ist, dass die Posenbestimmung im Idealfall sehr genau arbeitet und der modifizierte ICP-Algorithmus sehr gut funktioniert.

Bei genauer Betrachtung von Abbildung 5.8 sowie Abbildung 5.5 fällt auf, dass im Zeitraum 1 der Simulationen die Werte der Fehlermetrik $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ größer sind und mehr schwanken als im restlichen Zeitraum der Simulation. Um dies genauer zu untersuchen, wird das Ergebnis der Posenbestimmung $\mathbf{H}_{\text{icp}}$ vor der Filterung betrachtet, siehe Abschnitt 3.3.2. Die Drehung durch die Rotationsmatrix $\mathbf{R}_{\text{icp}}$ wird zu diesem Zweck zuerst auf das Sensorkoordinatensystem ${}_{\mathcal{S}}\mathbf{R}_{\text{icp}}$ bezogen und mithilfe der Funktion $f_{\text{x,y,z}}(\mathbf{R})$ auf drei

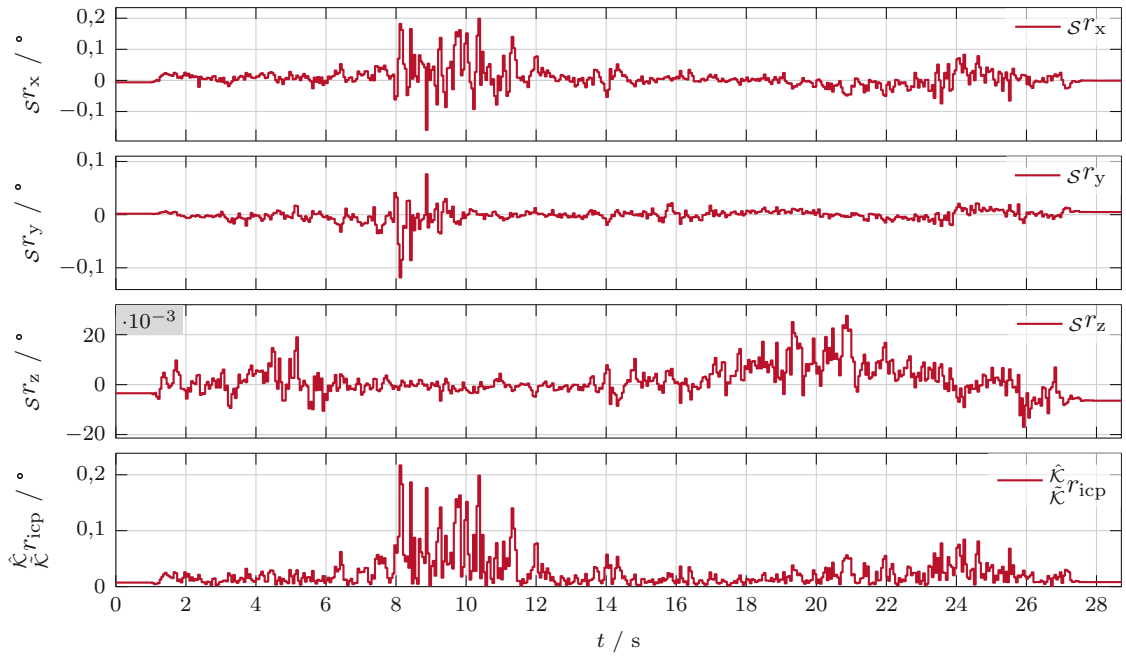


Abbildung 5.9: Die rotatorische Fehlermetrik der Posenbestimmung dargestellt als extrinsische Drehung um die x, y, z-Achse ohne Messrauschen und ohne Modellfehler.

Euler-Winkel gemäß

$$\mathcal{S}\mathbf{R}_\mathrm{icp} = f_\mathrm{R}\left(\mathcal{S}_\mathcal{O}\mathbf{H}\,\mathbf{H}_\mathrm{icp}\,{}^\mathcal{O}_\mathcal{S}\mathbf{H}\right) \tag{5.3a}$$

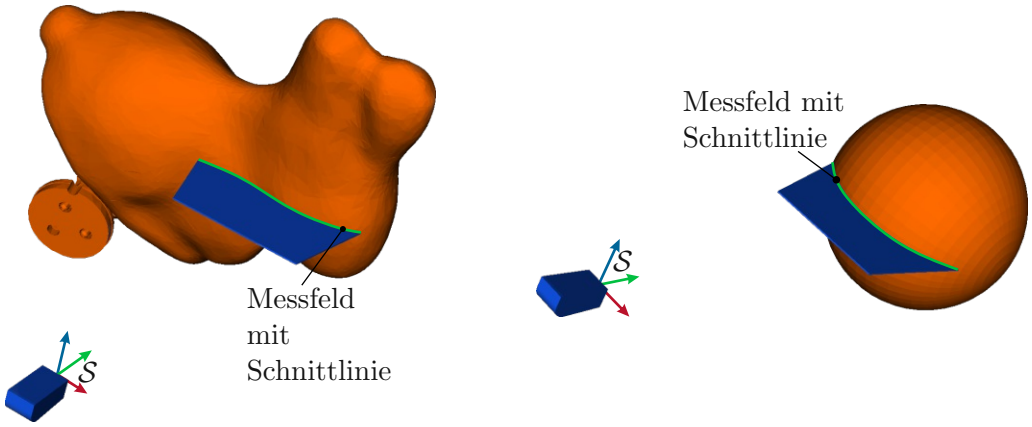
$$\begin{bmatrix}\mathcal{S}r_\mathrm{x} & \mathcal{S}r_\mathrm{y} & \mathcal{S}r_\mathrm{z}\end{bmatrix}^\mathrm{T} = f_\mathrm{x,y,z}(\mathcal{S}\mathbf{R}_\mathrm{icp})\ . \tag{5.3b}$$

aufgeteilt. Die Winkel $\mathcal{S}r_\mathrm{x}$, $\mathcal{S}r_\mathrm{y}$ und $\mathcal{S}r_\mathrm{z}$ beschreiben eine extrinsische Drehung zuerst um die x-Achse, die y-Achse und zuletzt die z-Achse. Für die Herleitung der Berechnung wird auf Siciliano et al. [18] verwiesen.

Der Betrag des Winkels der Rotation aus der Rotationsmatrix $\mathcal{S}\mathbf{R}_\mathrm{icp}$ ist äquivalent zu der Fehlermetrik der Rotation des Positionsfehlers $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r_\mathrm{icp}$, siehe Abschnitt 5.1.3.

In Abbildung 5.9 sind in den ersten drei Graphen die Drehungen um die x-Achse $\mathcal{S}r_\mathrm{x}$, y-Achse $\mathcal{S}r_\mathrm{y}$ und z-Achse $\mathcal{S}r_\mathrm{z}$ dargestellt, die Fehlermetrik der Rotation des Positionsfehlers $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r_\mathrm{icp}$ ist im letzten Graph abgebildet. Wenn die ersten drei Graphen mit der Fehlermetrik $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r_\mathrm{icp}$ verglichen werden, ist zu sehen, dass der Großteil der Rotation durch die Drehung um die x-Achse $\mathcal{S}r_\mathrm{x}$ entsteht.

Zum Zeitpunkt $t = 9\,\mathrm{s}$ stehen Sensor und das Objekt in einer Anordnung, sodass die Schnittlinie grob vereinfacht einer Linie parallel zur x-Achse des 2D-Lasertriangulationssensors entspricht, siehe Abbildung 5.10a. Dadurch enthalten die Messdaten zu wenig Information um die Drehung um die x-Achse exakt zu bestimmen. Dies führt zu einem größeren systematischen Fehler bei der Schätzung der korrekten Orientierung des Objektes.



(a) (b)

Abbildung 5.10: Darstellung der Ursachen für systematische Fehler in der Posenbestimmung, (a) Vermessung des Objektes durch den 2D-Lasertriangulationssensor zum Zeitpunkt $t = 9\,\mathrm{s}$, (b) Darstellung der Rotationssymmetrie an einer Kugel.

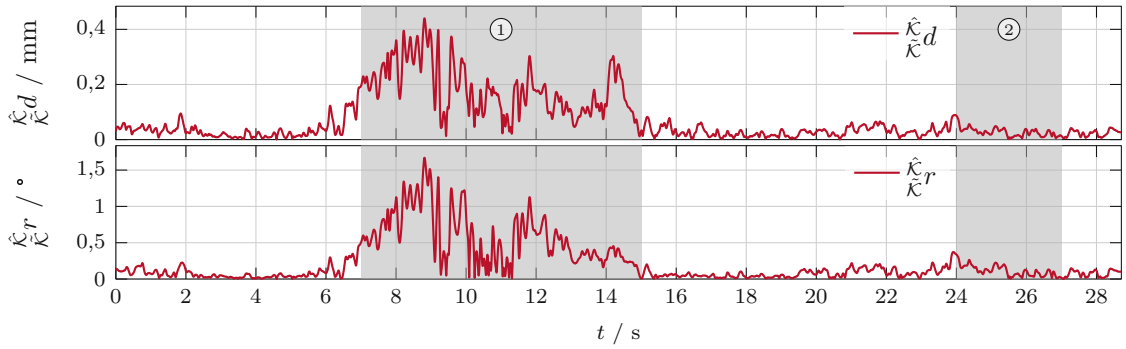


Abbildung 5.11: Fehlermetriken der Posenbestimmung mit aktiviertem Messrauschen und ohne Modellfehler.

Eine andere mögliche Ursache für einen systematischen Fehler in der Posenbestimmung ist, wenn die Oberfläche des Objektes an der Schnittlinie rotationssymmetrisch ist. In Abbildung 5.10b ist dieser Sachverhalt für den einfachen Fall einer Kugel dargestellt. Durch die Rotationssymmetrie ist es nicht möglich, die genaue Pose zu bestimmen. Dies führt zu einem zusätzlichen systematischen Fehler.

5.5.2 Simulation mit Messrauschen und ohne Modellfehler

Zur weiteren Untersuchung der Genauigkeit wird das Messrauschen des 2D-Lasertriangulationssensors aktiviert mit einer Standardabweichung von $\sigma_\mathrm{E} = 65\,\mu\mathrm{m}$. In Abbildung 5.11 sind die entsprechenden Fehlermetriken der Posenbestimmung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ dargestellt. Im Vergleich zu den selben Fehlermetriken $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ bei aktiviertem Messrauschen und

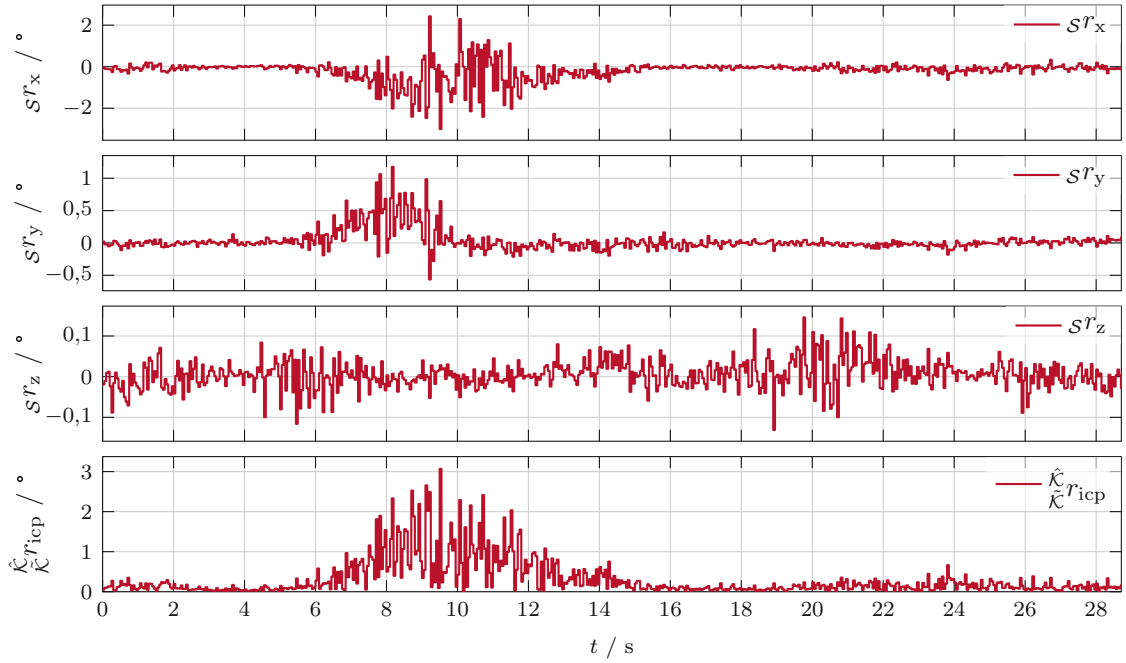


Abbildung 5.12: Die rotatorische Fehlermetrik der Posenbestimmung dargestellt als extrinsische Drehung um die x-, y- und z-Achse mit aktiviertem Messrauschen und ohne Modellfehler.

mit Modellfehler, siehe Abbildung 5.5, ist zu erkennen, dass die Fehlermetrik ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r$ sehr ähnlich ist. Bei beiden Graphen ist der maximale Wert 1,6°. Dagegen ist die translatorische Fehlermetrik der Posenbestimmung ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}d$ ohne Modellfehler deutlich kleiner. Im Zeitraum 1 ist der maximale translatorische Fehler um 0,2 mm niedriger, d. h. 0,4 mm anstatt 0,6 mm. Im Zeitraum 2 tritt bei der Simulation ohne Modellfehler kein zusätzlicher Fehler auf. Dementsprechend kann angenommen werden, dass der erhöhte Fehler in Abbildung 5.5 durch den Modellfehler hervorgerufen wird.

In Abbildung 5.12 ist die Rotation durch drei Winkel relativ zum Sensorkoordinatensystem $\mathcal{S}$ dargestellt, siehe (5.3). Zusätzlich ist die Fehlermetrik der Rotation des Positionsfehlers ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{icp}$ abgebildet. Analog zur Simulation ohne Messrauschen des Sensors entsteht der Großteil des Fehlers durch die Drehung um die x-Achse ${}_{\mathcal{S}}r_x$, vgl. Abbildung 5.9. Die Ursache für den systematischen Fehler ist derselbe wie davor, siehe Abschnitt 5.5.1 und durch das Messrauschen wird dieser Effekt weiter verstärkt. Dadurch steigt das Maximum der rotatorischen Fehlermetrik der Posenbestimmung ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{icp}$ von 0,2° auf 3°.

Ein Vergleich der Fehlermetrik vor der Filterung ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{icp}$ aus Abbildung 5.12 und der Fehlermetrik ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r$ aus Abbildung 5.11 zeigt, dass durch die Filterung des Ergebnisses der Posenbestimmung der maximale Fehler ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r$ von 3° auf 1,6° reduziert wird. Dieser Anteil des Fehlers in der Posenbestimmung entsteht durch die Geometrie der Oberfläche des Objektes im Zusammenhang mit der Pose des Sensors relativ zum Objekt. Dieser Anteil des Fehlers ist unabhängig vom Modellfehler des Roboters.

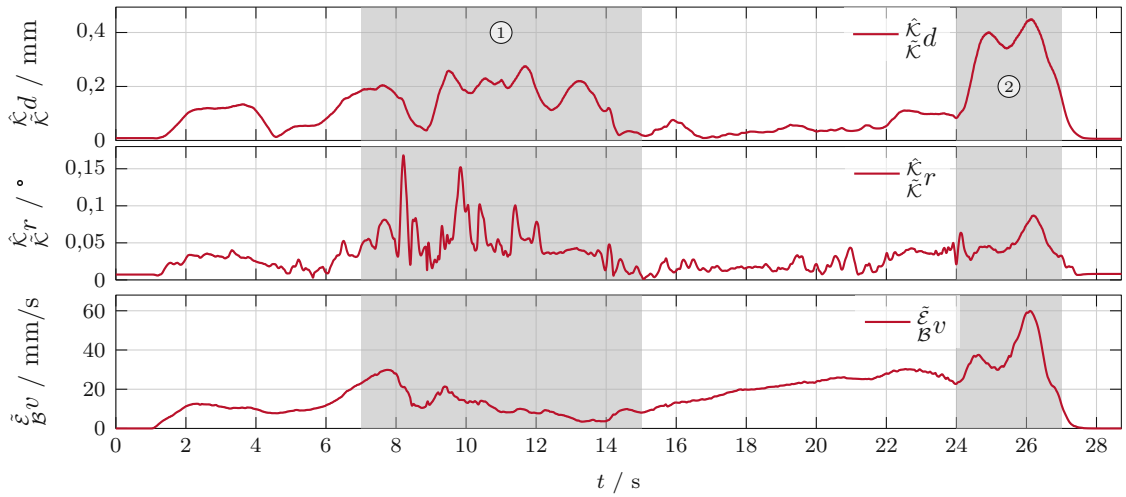


Abbildung 5.13: Fehlermetriken der Posenbestimmung sowie die Endeffektorgeschwindigkeit mit deaktiviertem Messrauschen und aktiviertem Modellfehler.

5.5.3 Simulation ohne Messrauschen und mit Modellfehler

Für die weitere Untersuchung der Genauigkeit wird in Abbildung 5.13 das Ergebnis der Simulation ohne Messrauschen des 2D-Lasertriangulationssensors, d. h. $\sigma_{\mathrm{E}} = 0\,\mu\mathrm{m}$, jedoch mit Modellfehler $\tilde{\mathbf{d}}_{\mathrm{y}}$ und $\tilde{\mathbf{q}}$ laut Tabelle 5.1, dargestellt. Der maximale Wert der rotatorischen Fehlermetrik der Posenbestimmung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ ist 0,17°. Dies ist gegenüber dem Wert von 1,6° bei der Simulation mit aktiviertem Messrauschen und ohne Modellfehler gering, vgl. Abbildung 5.11. Die translatorische Fehlermetrik der Posenbestimmung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ hat den maximalen Fehler im Zeitraum 2 mit einem Wert von 0,45 mm. Im Gegensatz dazu tritt bei der Simulation mit aktiviertem Messrauschen und ohne Modellfehler der maximale translatorische Fehler $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ im Zeitraum 1 auf und im Zeitraum 2 ist kein größerer Fehler ersichtlich, vgl. Abbildung 5.11. Daraus kann geschlossen werden, dass der Modellfehler andere Fehler in der Posenbestimmung verursacht als das Messrauschen.

Wenn die beiden Fehlermetriken $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ mit der Geschwindigkeit des Endeffektors $\tilde{\mathcal{E}}_{\mathcal{B}}v$ relativ zum Basiskoordinatensystems $\mathcal{B}$ verglichen werden, ist es ersichtlich, dass der Fehler der Posenbestimmung mit größerer Endeffektorgeschwindigkeit steigt. Der Grund dafür ist, dass der Modellfehler abhängig von den Gelenkwinkeln $\mathbf{q}$ ist und im Allgemeinen eine schnellere Bewegung des Endeffektors zu einer schnelleren Änderung der Gelenkwinkeln führt. Dadurch steigt die momentane Änderungsrate des Modellfehlers. Die Posenbestimmung hat eine diskretes Abtastzeit von $T_{\mathrm{icp}} = 50\,\mathrm{ms}$. Durch eine hohe Änderungsrate des Modellfehlers kommt es zu einem größerem Fehler in der Pose des Objektes innerhalb eines Abtastintervalls. Dieser Fehler ist unabhängig von dem zu bearbeitenden Werkstück, denn er entsteht durch die fehlerhafte Bewegung des Industrieroboters aufgrund des Modellfehlers.

Um diese Problematik genauer darzustellen, ist in Abbildung 5.14 der Zeitraum zwischen 25,8 s bis 26,5 s vergrößert dargestellt. In den oberen zwei Graphen ist die Fehlermetrik

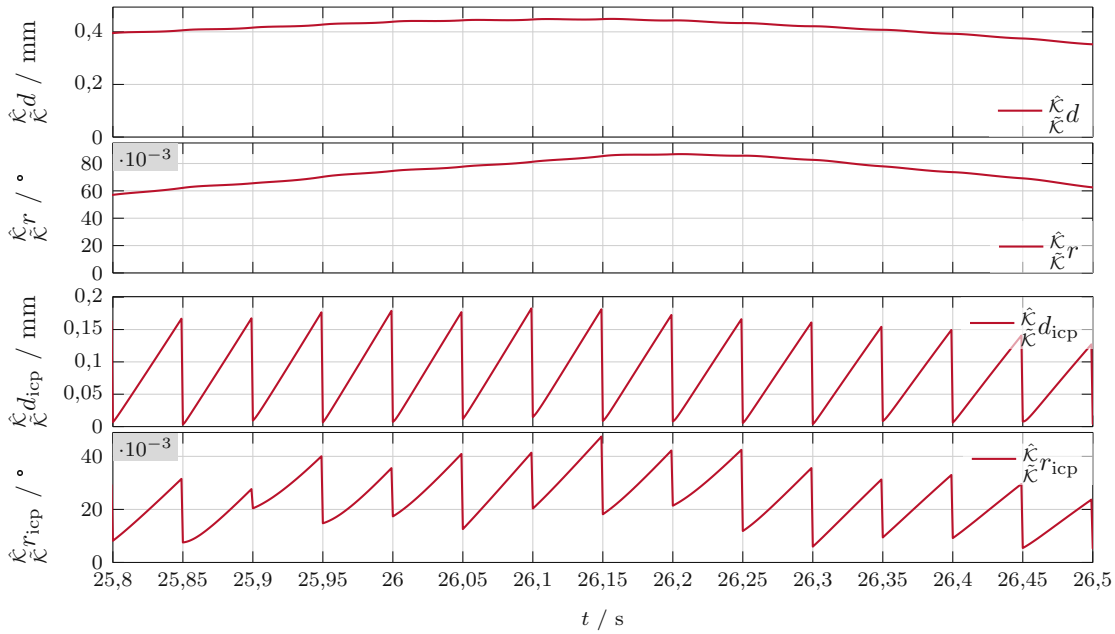


Abbildung 5.14: Vergrößerte Darstellung von Abbildung 5.13 im Zeitraum von 25,8 s bis 26,5 s.

der Posenbestimmung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ zu sehen. In den unteren zwei Graphen ist die Fehlermetrik der Posenbestimmung ohne Filterung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d_{\mathrm{icp}}$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r_{\mathrm{icp}}$ abgebildet. In regelmäßigen Intervallen mit $T_{\mathrm{icp}} = 50\,\mathrm{ms}$ erfolgt eine neue Posenbestimmung, wodurch dieser Fehler stark fällt und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d_{\mathrm{icp}}$ kleiner als 10 µm und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r_{\mathrm{icp}}$ kleiner als 20 Milligrad wird. Dies ist in Bezug auf die Genauigkeit sehr ähnlich zu den Ergebnissen der Simulation ohne Industrieroboter in Abschnitt 3.4, vgl. Tabelle 3.5. Zwischen den Abtastzeitpunkten bewegt sich der mit Modellfehlern behaftete Industrieroboter aber weiter, der durch den Modellfehler verursachte Fehler ändert sich und kann von der Posenbestimmung erst beim nächsten Abtastzeitpunkt korrigiert werden. Durch den Filter wird der Fehler weiter verzögert und geglättet, wodurch die Fehlermetriken $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}d$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}}r$ entstehen. Durch die Verzögerung steigt der maximale Wert in der Fehlermetrik von 0,16 mm auf 0,45 mm sowie 40 Milligrad auf 80 Milligrad. Der Filter für das Ergebnis der Posenbestimmung wird relativ langsam gewählt, um hohe Spitzen in den Drehmomenten und der Geschwindigkeit des Industrieroboters zu vermeiden.

Dieser Anteil des Fehlers entsteht durch die Zeitverzögerung, welche durch die diskrete Abtastzeit T_{icp} und dem langsamen Filter der Posenbestimmung entsteht. Dieser Anteil des Fehlers ist unabhängig von der Oberflächengeometrie des Werkstückes und von der Posenbestimmung.

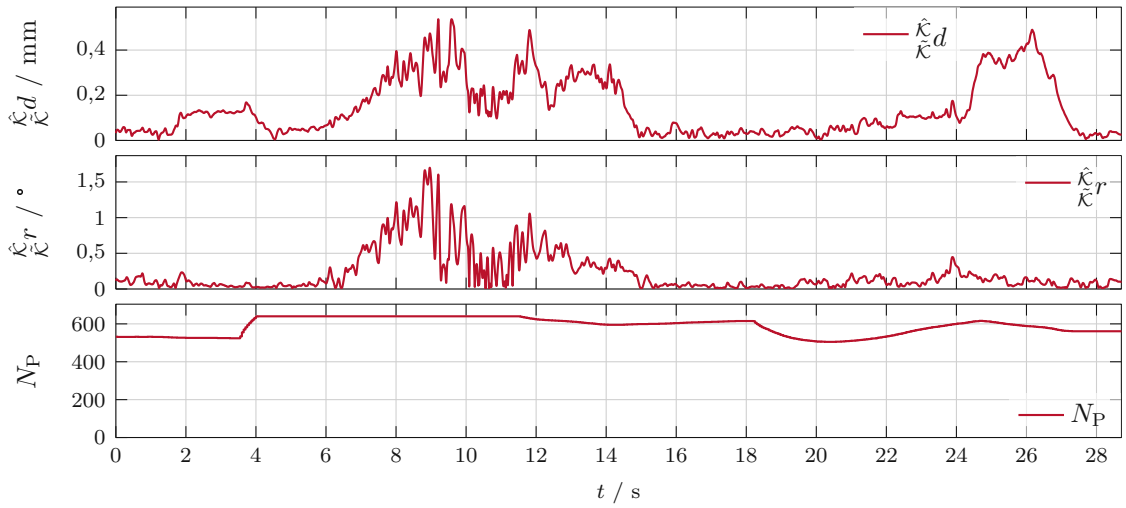


Abbildung 5.15: Vergleich der Anzahl der Messpunkte mit der Fehlermetrik der Posenbestimmung.

5.5.4 Simulation mit variierender Anzahl an Messpunkten

Aufgrund der Geometrie und der begrenzten Ausdehnung des Werkstückes treffen nicht immer alle Messpunkte des Messfeldes des 2D-Lasertriangulationssensors auf die Oberfläche des Werkstückes. In Abbildung 5.15 ist der Verlauf der Anzahl der Messpunkte N_P der Fehlermetrik der Posenbestimmung $\hat{\kappa}_{\tilde{\kappa}} d$ und $\hat{\kappa}_{\tilde{\kappa}} r$ gegenübergestellt. Dabei ist kein Zusammenhang zwischen der Genauigkeit und der Anzahl der gültigen Messpunkte festgestellt worden.

5.6 Untersuchung der Laufzeit der Posenbestimmung

In diesem Abschnitt wird die Laufzeit der Posenbestimmung untersucht. Für die Messung der Laufzeit des Algorithmus wird in MATLAB/SIMULINK der `Profiler Report` verwendet. Dieser misst die durchschnittlich benötigte Laufzeit pro Aufruf für jeden Block in der Simulation. Es wird die Laufzeit der Posenbestimmung untersucht, da in einem realen Aufbau sowohl der Sensor als auch der Roboter nicht simuliert werden müssen. Weiters ist die Laufzeit des Algorithmus von zentraler Bedeutung für die Implementierung auf echtzeitfähiger Hardware. Im ersten Schritt wird der Einfluss der Anzahl der Messstrahlen N_S auf die Laufzeit t_l, die Konvergenz der Posenbestimmung und die Genauigkeit der Posenbestimmung untersucht. Dafür wird die Simulation mit Messrauschen des 2D-Lasertriangulationssensors und dem Modellfehler durchgeführt, analog zu Abschnitt 5.3. In Tabelle 5.2 sind die Ergebnisse zusammengefasst. Die Laufzeit des Algorithmus t_l verringert sich erwartungsgemäß mit fallender Anzahl der Messpunkte N_P von 38 ms bis auf 6 ms. Durch eine verringerte Anzahl an Messpunkten benötigt der Algorithmus der Posenbestimmung durchschnittlich mehr Iterationen k_it für die Berechnung der Pose. Die durchschnittlich benötigten Iterationen k_it steigen dabei leicht von 1,92 auf 2,63.

Tabelle 5.2: Vergleich der Laufzeiten mit verschieden vielen Messpunkten.

Name Symbol Einheit	Messstrahlen N_S 1	Messpunkte mean N_P 1	Laufzeit mean t_l ms	Iterationen mean k_{it} 1	Abbrüche N_E 1
	640	589	38	1,92	0
	320	295	24	2,14	3
	160	147	11	2,41	7
	80	73	6	2,63	31

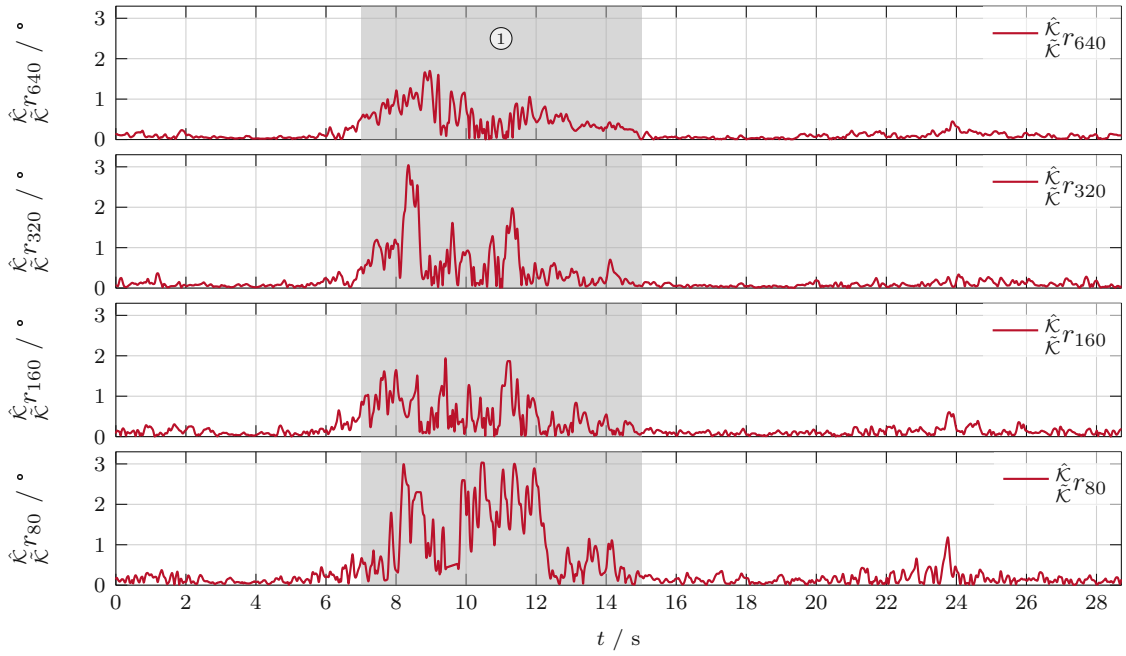


Abbildung 5.16: Vergleich der rotatorischen Fehlermetrik der Posenbestimmung für verschieden viele Messpunkte.

In Bezug auf die Konvergenz der Posenbestimmung wurde die Anzahl der Algorithmusabbrüche N_E untersucht. Dies tritt auf, wenn die geschätzte Pose die maximale Translation $\tau_{T,max}$ oder Rotation $\tau_{R,max}$ überschreitet, siehe (3.21). Mit fallender Anzahl an Messpunkten steigt die Wahrscheinlichkeit, dass die Posenbestimmung abgebrochen wird. Bei $N_S = 640$ kommt es zu keinem Abbruch, wohingegen es bei $N_S = 80$ zu 31 Abbrüchen während eines Durchlaufes kommt. Insgesamt gibt es bei einer Simulationsdauer von 28,7 s und einer Abtastzeit von $T_{icp} = 50\,\text{ms}$ 574 Durchläufe der Posenbestimmung.

Um die erreichte Genauigkeit zwischen den Simulationen mit unterschiedlich vielen Messstrahlen N_S zu vergleichen, wird in Abbildung 5.16 die rotatorischen Fehlermetriken der Posenbestimmung $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}} r_{640}$, $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}} r_{320}$, $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}} r_{160}$ und $\hat{\mathcal{K}}_{\tilde{\mathcal{K}}} r_{80}$ gegenübergestellt. Es wird die rotatorische Fehlermetrik verwendet, weil diese weniger vom Modellfehler des Roboters

beeinflusst wird.

Der maximale Wert der rotatorischen Fehlermetrik steigt von 1,8° bei ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{640}$ auf 3,1° bei ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{80}$. Wenn die Fehlermetriken ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{640}$, ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{320}$ und ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{160}$ ohne Zeitraum 1 betrachtet werden, ist nur ein leichter Anstieg des Fehlers mit fallender Messpunkte festzustellen. Dagegen weist die Fehlermetrik ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{80}$ einen deutlich höheren Fehler mit Spitzen über 1° auf. In Zeitraum 1 sind die Fehlermetriken bei verschiedener Anzahl der Messpunkte sehr unterschiedlich. Auf den ersten Blick sieht es so aus, als wäre die Fehlermetrik ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{160}$ genauer als die Fehlermetrik ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{320}$. Der Grund dafür ist die Spitze zum Zeitpunkt 8,5 s bei ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{320}$, welche bei ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{160}$ nicht auftritt. Der Unterschied entsteht durch die Abbruchbedingung, wenn ein maximaler Fehler überschritten wird, siehe (3.21). Diese Bedingung löst bei ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{320}$ noch nicht aus und bei ${}^{\hat{\mathcal{K}}}_{\tilde{\mathcal{K}}}r_{160}$ schon. In diesem Fall entsteht dadurch ein genaueres Resultat. Im Allgemeinen ist es jedoch besser, wenn die Posenbestimmung zu einem Ergebnis konvergiert anstatt abgebrochen zu werden.

Aus dieser Untersuchung kann festgehalten werden, dass das genaueste Ergebnis bei voller Anzahl der Messpunkte erreicht wird. Eine Verringerung der Messpunkte auf 320 oder 160 verursacht nur eine leichte Steigerung des Fehlers in der Posenbestimmung, hat dabei aber eine merklich kürzere Laufzeit.

5.7 Diskussion

Die Ergebnisse in Abschnitt 5.2 zeigen, dass bereits kleinste Änderungen $\tilde{\mathbf{d}}_{\mathrm{y}}$ und $\tilde{\mathbf{q}}$ in der direkten Kinematik einen großen Einfluss auf die Gesamtgenauigkeit des Roboters haben.

Die Genauigkeit der Posenbestimmung mit Rückführung zum Regler und die Gesamtgenauigkeit werden in Abschnitt 5.3 untersucht. Der translatorische Anteil der Fehlermetrik zeigt eine deutliche Verbesserung gegenüber der Simulation ohne Posenbestimmung, vgl. Abschnitt 5.2. Der rotatorische Anteil der Fehlermetrik zeigt eine Verringerung des Fehlers.

Abschnitt 5.5 beweist, dass die Modifizierungen des ICP-Algorithmus funktionieren und die gewählten Vereinfachungen und Approximationen einen vernachlässigbaren Einfluss auf die Genauigkeit haben. Es können zwei Ursachen für die variierende Genauigkeit der Posenbestimmung in Abschnitt 5.3 gefunden werden:

- Erstens entsteht ein systematischer Fehler aufgrund des Messrauschens des 2D-Lasertriangulationssensors und der Oberflächenform des Objektes. Dabei zeigte sich, dass dieser Fehler an Stellen des Werkstückes, an denen die Messpunkte näherungsweise eine Gerade bilden, größere Auswirkungen hat. An diesen Stellen ist es für die Posenbestimmung schwieriger die korrekte Rotation des Werkstückes zu berechnen. Durch das Messrauschen wird dieser Fehler verstärkt. Auch wurde gezeigt, dass durch eine Rotationssymmetrie der Oberfläche die Berechnung der Pose erschwert wird.

- Zweitens entsteht durch die diskrete Abtastzeit T_{icp} stets ein Fehler, weil sich der Modellfehler durch die Bewegung des Roboters im Zeitraum bis zum nächsten Abtastzeitpunkt ändert. Durch die zusätzliche Filterung des Ergebnisses der Posenbestimmung wird der Effekt verstärkt. Der Filter wird benötigt, um einen stetigen

Verlauf der Pose des Werkstückes zu erhalten.

Abschnitt 5.6 untersucht die Laufzeit der Posenbestimmung bei wechselnder Anzahl an Messpunkten N_S. Die Untersuchung zeigt, dass sich mit weniger Messpunkten die Laufzeit der Posenbestimmung deutlich verringert. Die erreichte Genauigkeit des Algorithmus ändert sich im Allgemeinen durch Reduktion der Messpunkte auf $N_S = 160$ nicht stark.

Die Ergebnisse in Abschnitt 5.4 decken auf, dass der PD-Regler für die Regelung eines Industrieroboters mit Modellfehler für diesen Anwendungsfall nicht geeignet ist. Die Simulationen zeigen hingegen die Eignung des PID-Reglers und dessen Stabilität.

6 Zusammenfassung und Ausblick

6.1 Zusammenfassung

Das Ziel der Forschungsarbeit war es zu zeigen, dass mithilfe eines ortsfesten 2D-Lasertriangulationssensors eine absolut genaue Bearbeitung eines Werkstückes durch einen unkalibrierten Industrieroboter ermöglicht werden kann. Um die Absolutgenauigkeit zu erreichen, wurde das Werkstück mit einem 2D-Lasertriangulationssensor vermessen und die Pose des Werkstückes durch einen angepassten Iterative-Closest-Point-Algorithmus (ICP-Algorithmus) berechnet. Die Algorithmen dieser Arbeit wurden anhand von Simulationen eines Arbeitsablaufes demonstriert.

In Kapitel 2 wurden die benötigten mathematischen Modelle hergeleitet. Im Speziellen wurde das Modell des Roboters KUKA LBR iiwa 14 R820 als Starrkörpermodell mit sieben Freiheitsgraden sowie ein Modell des 2D-Lasertriangulationssensors MICRO-EPSILON scanCONTROL 2600-100 hergeleitet. Das Sensormodell basiert auf der geometrischen Berechnung der Schnittpunkte zwischen den Laserlinien des Sensors und der diskreten Oberfläche des Werkstückmodells. Der Industrieroboter wurde mit den nominellen Parametern modelliert und zusätzlich wurden typische kinematische Fehler im Robotermodell eingeführt. Die Modellfehler wurden durch Ungenauigkeiten in der Geometrie des Roboters und der Gelenkwinkelsensoren abgebildet.

Die Posenbestimmung wurde in Kapitel 3 präsentiert. Diese beruht auf dem ICP-Algorithmus und vergleicht die Messpunkte mit der bekannten Oberfläche des Modells des Objektes, um die korrekte Pose zu ermitteln. Im Rahmen dieser Arbeit wurden eine Reihe von Änderungen und Verbesserungen des Iterative-Closest-Point-Algorithmus untersucht. Zunächst wurde die Punkt-zu-Punkt-Fehlermetrik auf eine Punkt-zu-Ebenen-Fehlermetrik geändert. Mithilfe einer Approximation für kleine Winkel konnte das nichtlineare Optimierungsproblem auf eine lineare Optimierung nach dem kleinsten Quadrat vereinfacht werden. Zusätzlich wurde durch eine Vereinfachung der Suche des am nächsten liegenden Punktes die benötigte Laufzeit des Algorithmus von 116 s auf 23 ms reduziert ohne Einschränkung bei der erreichten Genauigkeit der Posenbestimmung. Um einen stetigen Verlauf der Pose zu erhalten, wurde die berechnete Pose mithilfe eines Filters geglättet. In einer Simulation konnte die verbesserte Genauigkeit und die schnellere Laufzeit gegenüber des ursprünglichen Algorithmus gezeigt werden.

Die Trajektorienplanung und die Regelung des Industrieroboters wurden in Kapitel 4 beschrieben. Die Regelung erfolgt mithilfe der inversen Dynamik in kartesischen Koordinaten relativ zum Objektkoordinatensystem des Werkzeuges. Sechs der sieben Freiheitsgrade des Roboters wurden durch die Trajektorie vorgegeben. Um den siebten Freiheitsgrad zu stabilisieren, wurde ein zusätzlicher Nullraumregler eingeführt. Damit das Ergebnis der Posenbestimmung in der Regelung verwendet werden konnte, wurde dieses mit dem Systemausgang des nominellen Modells des Roboters kombiniert. Für die Regelung wurde

eine zweifach stetig differenzierbare Trajektorie auf dem Objekt vorgegeben.

Die Ergebnisse der Simulations- und Parameterstudie wurden in Kapitel 5 präsentiert. Dabei wurde zuerst eine Simulation mit deaktivierter Posenbestimmung durchgeführt. Diese zeigt, dass Ungenauigkeiten in der Geometrie von 2 mm und ein Fehler in den Drehgebern von maximal 1° im Modell des Roboters zu hohen Abweichungen von der vorgegebenen Trajektorie führt. Die maximale Abweichung betrug dabei translatorisch 14 mm und rotatorisch 2,6°. Auch wurde die vorgegebene Pfadgeschwindigkeit von 7 mm/s nicht eingehalten. Durch die Verwendung der Posenbestimmung konnte die maximale Abweichung auf translatorisch 0,6 mm und rotatorisch auf 1,5° reduziert werden. Die Pfadgeschwindigkeit wurde eingehalten. Weiters wurden die Ursachen für den Restfehler untersucht. Dabei konnte der Hauptteil der Abweichung auf zwei Ursachen zurückgeführt werden. Einerseits ergeben sich während der Simulation Posen, an denen die Messpunkte des 2D-Lasertriangulationssensors keine eindeutige Berechnung der exakten Posen des Werkstückes erlauben. Andererseits entsteht durch die diskreten Abtastzeiten der Posenbestimmung und der Zeitverzögerung des Filters Abweichungen, welche erst zeitverzögert erkannt und korrigiert werden können.

Schließlich wurde die Laufzeit der Posenbestimmung für verschiedene Einstellungen untersucht und verglichen. Mit der erreichten Laufzeit ist ein Einsatz in einem realen Aufbau als echtzeitfähige Implementierung möglich.

6.2 Ausblick

Mit den Ergebnissen der Simulationen in Kapitel 5 wurde gezeigt, dass die Absolutgenauigkeit im geschlossenen Regelkreis erreicht werden kann.

Im nächsten Schritt kann dieses Konzept in einem realen Aufbau mit einem Industrieroboter und einem 2D-Lasertriangulationssensor implementiert und getestet werden. Um die Genauigkeit der Posenbestimmung noch weiter zu verbessern, ergeben sich aus der Untersuchung der Genauigkeit in Abschnitt 5.5 noch folgende Forschungsrichtungen:

- Diese Arbeit zeigt, dass die erreichbare Genauigkeit der Posenbestimmung schwankt bzw. die Pose in manchen Situationen nicht exakt bestimmt werden kann. Kann die Robustheit und die Genauigkeit durch eine veränderte Positionierung des Sensors verbessert werden?

- Der Filter der Posenbestimmung hat eine relativ langsame Zeitkonstante um die Änderungsrate zu begrenzen. Kann der Fehler aufgrund der Zeitverzögerung durch ein verbessertes Filterkonzept minimiert werden?

- Der Nullraumregler wird in dieser Arbeit verwendet, um die Gelenkwinkel möglichst nahe beim Nullpunkt zu halten. Kann die Winkelgeschwindigkeit der Gelenke durch einen angepassten Nullraumregler minimiert werden, um die Änderungsrate des Modellfehlers minimal zu halten?

A Anhang

A.1 Posen der Bestandteile der Simulation

Die Beschreibung der Pose der Basis des Roboters $\mathcal{B}$ in Bezug auf das Weltkoordinatensystem $\mathcal{W}$ ergibt sich mithilfe der homogenen Transformation ${}^{\mathcal{B}}_{\mathcal{W}}\mathbf{H}$ gemäß

$$ {}^{\mathcal{B}}_{\mathcal{W}}\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 1.9 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{A.1} $$

Die Beschreibung der Pose des Werkzeugs $\mathcal{T}$ in Bezug auf das Weltkoordinatensystem $\mathcal{W}$ ergibt sich mithilfe der homogenen Transformation ${}^{\mathcal{T}}_{\mathcal{W}}\mathbf{H}$ gemäß

$$ {}^{\mathcal{T}}_{\mathcal{W}}\mathbf{H} = \begin{bmatrix} 0 & 0 & -1 & -0.65 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1.5 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{A.2} $$

Die Beschreibung der Pose des 2D-Lasertriangulationssensors $\mathcal{S}$ in Bezug auf das Weltkoordinatensystem $\mathcal{W}$ ergibt sich mithilfe der homogenen Transformation ${}^{\mathcal{S}}_{\mathcal{W}}\mathbf{H}$ gemäß

$$ {}^{\mathcal{S}}_{\mathcal{W}}\mathbf{H} = \begin{bmatrix} 0 & 0.866 & 0.5 & -0.8578 \\ -1 & 0 & 0 & 0 \\ 0 & -0.5 & 0.866 & 1.635 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{A.3} $$

Literatur

[1] A. Valente, S. Baraldo und E. Carpanzano, „Smooth trajectory generation for industrial robots performing high precision assembly processes“, *CIRP Annals*, Jg. 66, Nr. 1, S. 17–20, 2017.

[2] J. Norberto Pires, T. Godinho und P. Ferreira, „CAD interface for automatic robot welding programming“, *Industrial Robot: An International Journal*, Jg. 31, Nr. 1, S. 71–76, 2004.

[3] J. De Backer, „Robotic Friction Stir Welding for Flexible Production“, Diss., Lund University, Sweden, 2012. Adresse: `https://lucris.lub.lu.se/ws/portalfiles/portal/5489675/2856190.pdf` (besucht am 05. 11. 2021).

[4] H. Chen, T. Fuhlbrigge und X. Li, „Automated Industrial Robot Path Planning for Spray Painting Process: A Review“, in *IEEE Conference on Automation Science and Engineering*, Washington DC, USA, 2008, S. 522–527.

[5] M. Amersdorfer, J. Kappey und T. Meurer, „Real-time freeform surface and path tracking for force controlled robotic tooling applications“, *Robotics and Computer-Integrated Manufacturing*, Jg. 65, S. 1–15, 2020.

[6] W. Holub, F. Brunner und T. Schön, „RoboCT - Application for in-situ Inspection of Join Technologies of large scale Objects“, in *International Symposium on Digital Industrial Radiology and Computed Tomography*, Fürth, Germany, 2019, S. 1–7.

[7] M. Damak, J. Grosbois und P. De Smet, „Vision Robot based Absolute Accuracy Measurement“, in *International Symposium on Robotics*, Paris, France, 2004, S. 1–6.

[8] B. Greenway, „Robot accuracy“, *Industrial Robot: An International Journal*, Jg. 27, Nr. 4, S. 257–265, 2000.

[9] J. Qin, F. Léonard und G. Abba, „Real-Time Trajectory Compensation in Robotic Friction Stir Welding Using State Estimators“, *IEEE Transactions on Control Systems Technology*, Jg. 24, Nr. 6, S. 2207–2214, 2016.

[10] G. Boyé de Sousa, A. Olabi, J. Palos und O. Gibaru, „3D metrology using a collaborative robot with a laser triangulation sensor“, *Procedia Manufacturing*, Jg. 11, S. 132–140, 2017.

[11] M. Płaczek und Ł. Piszczek, „Testing of an industrial robot's accuracy and repeatability in off and online environment“, *Eksploatacja i Niezawodnosc*, Jg. 20, Nr. 3, S. 455–464, 2018.

[12] M. Morozov u. a., „Assessing the Accuracy of Industrial Robots through Metrology for the Enhancement of Automated Non-Destructive Testing“, in *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Baden, Germany, 2016, S. 335–340.

[13] ISO 9283:1998, „Manipulating industrial robots - Performance criteria and related test methods“, International Organization for Standardization, Geneva, CH, Techn. Ber., 1998. Adresse: `https://www.iso.org/standard/22244.html` (besucht am 05. 11. 2021).

[14] M. Slamani, A. Nubiola und I. Bonev, „Assessment of the positioning performance of an industrial robot“, *Industrial Robot: An International Journal*, Jg. 39, Nr. 1, S. 57–68, 2012.

[15] S. Gharaaty, T. Shu, A. Joubair, W. F. Xie und I. A. Bonev, „Online pose correction of an industrial robot using an optical coordinate measure machine system“, *International Journal of Advanced Robotic Systems*, Jg. 15, Nr. 4, S. 1–16, 2018.

[16] D. Aristos und S. Tzafestas, „A method for the registration of a known CAD model into the workspace frame of a robot“, *International Journal on Artificial Intelligence Tools*, Jg. 19, Nr. 03, S. 281–304, 2010.

[17] O. Aichholzer und B. Jüttler, *Einführung in die angewandte Geometrie*. Switzerland: Birkhäuser Basel, 2014.

[18] B. Siciliano, L. Sciavicco, L. Villani und G. Oriolo, *Robotics: Modelling, Planning and Control*. United Kingdom: Springer, 2010.

[19] H. Hahn, *Rigid Body Dynamics of Mechanisms: 1 Theoretical Basis*. Germany: Springer, 2002.

[20] C. Gaz, F. Flacco und A. De Luca, „Identifying the Dynamic Model Used by the KUKA LWR: A Reverse Engineering Approach“, in *IEEE International Conference on Robotics and Automation*, Hong Kong, China, 2014, S. 1386–1392.

[21] ——, „Extracting Feasible Robot Parameters from Dynamic Coefficients using Nonlinear Optimization Methods“, in *IEEE International Conference on Robotics and Automation*, Stockholm, Sweden, 2016, S. 2075–2081.

[22] MICRO-EPSILON Messtechnik GmbH & Co KG, *scanCONTROL 2D/3D Laser-Scanner (Laser-Profil-Sensoren), scanCONTROL 26x0*. Adresse: `https://www.micro-epsilon.com/download/manuals/man--scanCONTROL-26xx--de.pdf` (besucht am 05. 11. 2021).

[23] P. J. Besl und N. D. McKay, „Method for Registration of 3-D Shapes“, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Bd. 14, 1992, S. 239–256.

[24] B. K. P. Horn, H. M. Hilden und S. Negahdaripour, „Closed-form solution of absolute orientation using orthonormal matrices“, *Journal of the Optical Society of America A*, Jg. 5, S. 1127–1135, 1988.

[25] K.-L. Low, „Linear Least-Squares Optimization for Point-to-Plane ICP Surface Registration“, Department of Computer Science: University of North Carolina at Chapel Hill, North Carolina, USA, Techn. Ber., 2004. Adresse: `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.116.7292&rep=rep1&type=pdf` (besucht am 05. 11. 2021).

[26] W. Li und P. Song, „A modified ICP algorithm based on dynamic adjustment factor for registration of point cloud and CAD model“, *Pattern Recognition Letters*, Jg. 65, S. 88–94, 2015.

[27] J. B. Kuipers, *Quaternions and Rotation Sequences: A Primer with Applications to Orbits, Aerospace, and Virtual Reality*. United Kingdom: Princeton University Press, 1999.

[28] J. H. Friedman, J. L. Bentley und R. A. Finkel, „An Algorithm for Finding Best Matches in Logarithmic Expected Time“, *ACM Transactions on Mathematical Software*, Jg. 3, Nr. 3, S. 209–226, 1977.

[29] E. B. Dam, M. Koch und M. Lillholm, „Quaternions, Interpolation and Animation“, "Department of Computer Science: University of Copenhagen", Copenhagen, Denmark, Techn. Ber., 1998. Adresse: `https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.30.6788&rep=rep1&type=pdf` (besucht am 05. 11. 2021).

[30] N. Dantam, „Quaternion Computation“, Institute for Robotics und Intelligent Machines, Georgia Institute of Technology, Atlanta, USA, Techn. Ber., 2014. Adresse: `https://www.researchgate.net/publication/297038185_Quaternion_Computation` (besucht am 05. 11. 2021).

[31] C. Chen, D. Chen und K. Firm, *Linear System Theory and Design*. United Kingdom: Oxford University Press, 1999.

[32] KUKA Roboter GmbH, *LBR iiwa 14 R820*, 2017. Adresse: `https://www.kuka.com/-/media/kuka-downloads/imported/6b77eecacfe542d3b736af377562ecaa/db_lbr_iiwa_en.pdf` (besucht am 05. 11. 2021).

[33] C. Hartl-Nesic, *Surface-Based Path Following Control on Freeform 3D Objects*. Germany: Shaker Verlag GmbH, 2020. Adresse: `https://repositum.tuwien.at/handle/20.500.12708/15542` (besucht am 05. 11. 2021).

[34] J. S. C. Yuan, „Closed-Loop Manipulator Control Using Quaternion Feedback“, *IEEE Journal on Robotics and Automation*, Jg. 4, Nr. 4, S. 434–440, 1988.

[35] C. Hartl-Nesic, T. Glück und A. Kugi, „Surface-Based Path Following Control: Application of Curved Tapes on 3-D Objects“, *IEEE Transactions on Robotics*, Jg. 37, Nr. 2, S. 615–626, 2021.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit gemäß dem Code of Conduct – Regeln zur Sicherung guter wissenschaftlicher Praxis (in der aktuellen Fassung des jeweiligen Mitteilungsblattes der TU Wien), insbesondere ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel, angefertigt wurde. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In– noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Wien, im November 2021

Franz Stübler