



**FH Salzburg**

# **BACHELORARBEIT 1**

**Entwicklung einer Methodik zur Zusammenführung von Bildern  
zweier 3D-Kameras  
in Mensch-Roboter-Arbeitsplätzen**

durchgeführt am Studiengang  
Informationstechnik und System-Management  
Fachhochschule Salzburg GmbH

vorgelegt von  
Armin Niedermüller  
Christoph Untner  
Alexander Meier

Studiengangsleiter: FH-Prof. DI Dr. Gerhard Jöchl  
Betreuer: FH-Prof. DI (FH) DI Dr. Andreas Unterweger

Salzburg, Februar 2020

---

## Eidesstattliche Erklärung

Wir erklären hiermit eidesstattlich, dass wir die vorliegende Bachelorarbeit selbständig und ohne fremde Hilfe verfasst, und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben. Weiters versichern wir hiermit, dass wir die den benutzten Quellen wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht haben.

Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungskommission weder im In- noch im Ausland vorgelegt und auch nicht veröffentlicht.

Salzburg, am 16. Februar 2020

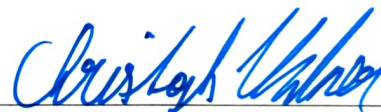
Ort, Datum



Armin Niedermüller

Salzburg, am 16. Februar 2020

Ort, Datum



Christoph Untner

Salzburg, am 16. Februar 2020

Ort, Datum



Alexander Meier

## **Abstract**

In smart factories, humans interact and work together with intelligent robots. Visual information is needed for the robot to safely and efficiently interact with humans. To provide an unobstructed view of the working environment, data from multiple cameras must be merged. This bachelor's thesis defines a process using Iterative-Closest-Point (ICP) and the required preprocessing to generate a transformation that can be used to align two 3-D video feeds.

This thesis discusses the fundamentals of the ICP, its variants and the applied preprocessing. It defines datasets which represent the expected environment in the workspace and uses these datasets to test the suitability of the different ICP variants. Additionally, the impact of the different preprocessing steps is evaluated.

The tests show that the nonlinear and the generalized ICP work well with the defined datasets, while the regular ICP is less robust and yields incorrect results for some datasets. For the process to reliably produce usable results, the preprocessing of the point clouds must perform a coarse pre-alignment, downsampling and restriction of the data so that the majority of surfaces are observed by both cameras.

---

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Problemstellung . . . . .	1
1.2	Zielsetzung . . . . .	2
1.3	Aufbau der Arbeit . . . . .	3
<b>2</b>	<b>Theoretische Grundlagen</b>	<b>4</b>
2.1	Stand der Forschung . . . . .	4
2.2	Kameraaufbau . . . . .	5
2.3	Koordinatentransformationen . . . . .	6
2.4	Grundlegende Vorgehensweise des ICP . . . . .	7
2.4.1	Bestimmung der Punkt-Korrespondenzen . . . . .	8
2.4.2	Minimierung der Fehlerfunktion . . . . .	8
2.4.3	Singular-Value-Decomposition . . . . .	10
2.5	Variationen des ICP . . . . .	12
2.5.1	Nonlinear-ICP . . . . .	12
2.5.2	Generalized-ICP . . . . .	12
2.6	Vorverarbeitung von Punktwolken . . . . .	15
2.6.1	Gründe für die Vorverarbeitung . . . . .	15
2.6.2	Angewandte Verfahren der Vorverarbeitung . . . . .	17
<b>3</b>	<b>Praktischer Teil</b>	<b>23</b>
3.1	Kamerahardware und Versuchsaufbau . . . . .	23
3.1.1	PC-Plattform . . . . .	24
3.2	Implementierung . . . . .	25
3.2.1	Verwendete Werkzeuge . . . . .	25
3.2.2	Testapplikation . . . . .	25
3.2.3	Prototyp . . . . .	26
3.3	Verwendete 3D-Datensätze . . . . .	27
3.4	Tests . . . . .	28
3.4.1	Auswahl der ICP-Variante . . . . .	28

3.4.2	Auswirkungen durch Deaktivierung einzelner Vorverarbeitungsschritte . . . . .	28
3.4.3	Parameterveränderung der einzelnen Vorverarbeitungsschritte . . .	30
<b>4</b>	<b>Fazit und Ausblick</b>	<b>35</b>
	<b>Anhang</b>	<b>36</b>
<b>A</b>	<b>Messergebnisse</b>	<b>36</b>
A.1	ICP-Varianten unter Verwendung aller Vorverarbeitungsschritte . . . . .	36
A.2	NL-ICP bei deaktivierten Vorverarbeitungsschritten . . . . .	40
A.3	NL-ICP bei optimierten Vorverarbeitungsschritten . . . . .	45

## Abkürzungsverzeichnis

**C-ICP** Constrained-ICP

**CoM** Center of Mass

**CPU** Central Processing Unit

**G-ICP** Generalized-ICP

**GPU** Graphics Processing Unit

**ICP-PTS** Point-To-Surface ICP

**ICP** Iterative-Closest-Point

**MLS** Moving Least Squares

**NL-ICP** Nonlinear-ICP

**OS** Operating System

**PCL** Point Cloud Library

**PT-Filter** Pass Through Filter

**RAM** Random-Access Memory

**ROS** Robot Operating System

**SOR-Filter** Statistical Outlier Removal Filter

**SVD** Singular-Value-Decomposition

**SVD-ICP** Singular-Value-Decomposition ICP

**VG-Filter** Voxel Grid Filter

## Abbildungsverzeichnis

Kameraaufbau . . . . .	5
Übersicht der einzelnen Schritte des ICP, adaptiert aus [4] . . . . .	7
Wahl der Punkt-Korrespondenzen über die nächsten Punkte, adaptiert aus [4] . . . . .	9
Huber-Verlustfunktion . . . . .	13
Schematische Darstellung einer Punkt-zu-Punkt Registrierung . . . . .	13
Schematische Darstellung einer Registrierung mit G-ICP . . . . .	14
Grobgranulare Betrachtung der unterschiedlichen Arbeitsschritte einer Punktwolken- Registrierung. . . . .	15
Die vier unabhängigen Verfahren, die in Summe die gesamte Vorverarbeitung bilden.	17
Die grünen und roten Punkte repräsentieren die Punkte einer Punktwolke, vor und nach dem Durchgangsfiler. . . . .	18
Schematische Darstellung des VG-Filter basierend auf [18]. . . . .	20
Verwendete Kamera- und Montagehardware . . . . .	23
Mensch-Roboter-Arbeitsplatz . . . . .	24
Schematische Darstellung des Ablaufs der Testapplikation . . . . .	26
Punktwolken nach der groben Vorausrichtung . . . . .	26
Ergebnis einer erfolgreichen Registrierung durch den NL-ICP . . . . .	27
Gemittelter Fitness-Score der ICP Varianten im Vergleich . . . . .	29
Auswirkungen des optimierten Parameters für VG Filter auf Fitness-Score und Ite- rationen . . . . .	31
Auswirkungen des optimierten Parameters für SOR Filter auf Fitness-Score und Iterationen . . . . .	32
Auswirkungen des optimierten Parameters für MLS auf Fitness-Score und Iterationen	33
Auswirkungen der optimierten Parameter aller Vorverarbeitungsschritte auf Fitness- Score und Iterationen . . . . .	34

## Tabellenverzeichnis

Registrierergebnisse nach Deaktivierung einzelner Vorverarbeitungsschritte . . . . .	29
Ergebnisse des Standard-ICP unter Verwendung aller Vorverarbeitungsschritte . . .	37
Ergebnisse des NL-ICP unter Verwendung aller Vorverarbeitungsschritte . . . . .	38
Ergebnisse des G-ICP unter Verwendung aller Vorverarbeitungsschritte . . . . .	39
Ergebnisse des NL-ICP ohne grobe Vorausrichtung . . . . .	40
Ergebnisse des NL-ICP ohne PT-Filter . . . . .	41
Ergebnisse des NL-ICP ohne VG-Filter . . . . .	42
Ergebnisse des NL-ICP ohne SOR-Filter . . . . .	43
Ergebnisse des NL-ICP ohne MLS . . . . .	44
Ergebnisse des NL-ICP mit optimierten VG-Filter . . . . .	45
Ergebnisse des NL-ICP mit optimierten SOR-Filter . . . . .	46
Ergebnisse des NL-ICP mit optimierten MLS . . . . .	47
Ergebnisse des NL-ICP mit allen Optimierungen der Vorverarbeitung . . . . .	48



# 1 Einleitung

Beginnend mit der Problemstellung in Abschnitt 1.1, führt Abschnitt 1.2 zu den geplanten Zielen dieser Arbeit. Abschnitt 1.3 erläutert den schriftlichen Aufbau des vorliegenden Schriftstücks.

## 1.1 Problemstellung

Innerhalb moderner Fabriken interagieren und arbeiten Menschen mit intelligenten Robotersystemen zur Reduzierung von Personalkosten und der Erhöhung der (allgemeinen) Produktivität. Mithilfe von 3D-Kameras kann es jenen teil-autonomen Maschinen ermöglicht werden, Gefahren für Mensch und Inventar zu erkennen. Daraus erfolgt ein erhöhtes Maß an Kontrolle und Sicherheit innerhalb der Arbeitsumgebung.

Defizitär an 3D-Kameras ist jedoch deren begrenztes optisches Sichtfeld, wodurch dem Roboter nur Teilbereiche der Arbeitsumgebung zur Verfügung stehen. Eine weitere Problematik stellt die Verdeckung von Bereichen hinter diversen stationären und instationären Objekten dar. Die Lösung dieser Einschränkungen kann mithilfe mehrerer, örtlich versetzter und unabhängiger 3D-Kameras erfolgen.

In der Robotik haben sich 3D-Kameras und sogenannte Light Detection and Ranging (LIDAR) Systeme etabliert, um Punktwolken zu erzeugen. Während 3D-Kameras dreidimensionale Bilder über stereoskopische oder „Time of Flight“-Verfahren erstellen, werden bei LIDAR Laserstrahlen sowie deren Reflexion und Lichtlaufzeit als Grundlage verwendet. 3D-Kameras bieten bei geringen Kosten eine gute Aufnahmequalität, während LIDARs in deutlich höheren Preisregionen anzutreffen sind. Sie besitzen einen großen Blickwinkel und können zusätzlich Farbinformationen aufnehmen.

Der in dieser Arbeit gewählte Testaufbau besteht aus zwei 3D-Kameras, welche über einem Mensch-Roboter-Arbeitsplatz montiert werden. Um eine allzeit „schattenfreie“ Erfassung des Arbeitsbereiches von etwa 1,5 m über und 0,5 m seitlich des Tisches zu gewährleisten, wird ein geeigneter Blickwinkel und Montageabstand der Kameras zueinander benötigt.

Zwei unabhängige 3D-Bilder beider Kameras müssen anschließend so zusammengesetzt werden, dass der Arbeitsplatz vollständig visuell erkennbar ist. Dazu muss zunächst die relative Position der 3D-Bilder zueinander bestimmt werden, um diese anschließend durch eine Drehung und Verschiebung auszugleichen. Durch eine Live-Aufnahme des gesamten Bereichs können Hindernisse als 3D-Objekte innerhalb der 3D-Bilder erkannt werden. Beachtet man diese Hindernisse bereits in der Bewegungsplanung des Roboters, kann dieser Menschen erkennen und unerwünschten Kontakt vermeiden. Die präzise Ausrichtung der Bilder sowie der zeitliche Aspekt deren Verarbeitung spielen deshalb eine essentielle

Rolle zur ganzheitlichen Erfassung des Arbeitsbereiches. Des Weiteren muss der Ansatz eine bestimmte Geschwindigkeit in seiner Verarbeitung aufweisen, um in weiterführenden Arbeiten eine schnellstmögliche Reaktion des Roboters zu gewährleisten.

3D-Bilder sind zudem als „Punktwolken“ bekannt und der Vorgang des Zusammensetzens mehrerer 3D-Bilder wird als „Punktwolken-Registrierung“ bezeichnet. Das aus den zwei Einzelbildern zusammengesetzte Bild wird folgend „Gesamtbild“ genannt.

Diese Arbeit beschreibt eine Methode zur Registrierung der Punktwolken für den oben beschriebenen Aufbau. Der de-facto Standard zur Punktwolken-Registrierung ist der sogenannte Iterative-Closest-Point (ICP).

## 1.2 Zielsetzung

Ziel der vorliegenden Arbeit ist, die beiden Punktwolken mittels ICP zu überlagern und ein Gesamtbild des Mensch-Roboter-Arbeitsplatzes zu realisieren. Um keine Gefahr für Mensch oder Inventar darzustellen, soll sich das erzeugte Gesamtbild dazu eignen, Hindernisse ausreichend genau und schnell zu erfassen. Dies bedeutet, dass der Roboter in weiterführenden Arbeiten in der Lage sein soll, vor einem beweglichen oder stationären Hindernis im Arbeitsraum stehen zu bleiben, wie beispielsweise der Kopf eines Menschen, welcher in den Arbeitsraum zu weit eindringt, während sich der Roboter bewegt.

Die Wirksamkeit bestehender Registrier-Algorithmen hängt stark von der Qualität der 3D-Daten ab. Dadurch kann es notwendig sein, die Punktwolken mit verschiedenen Methoden vorzuverarbeiten.

Es ergeben sich die folgenden Teilziele der Entwicklung einer Methodik zur Lösung der Problemstellung:

### 1) Bestmögliche Qualität der Punktwolken durch Vorverarbeitung

Anwenden geeigneter Filter auf die vorhandenen 3D-Datensätze. Dies kann die Ergebnisse der Registrier-Algorithmen verbessern. Die Qualität kann durch Verunreinigungen wie beispielsweise Signalrauschen oder Punktausreißer stark schwanken. Zur Messung der Überdeckungsqualität der Punktwolken-Registrierung wird die mittlere euklidische Distanz (siehe Abschnitt 2.1) beider Punktwolken verwendet.

### 2) Bestmögliche Qualität der Zusammenführung beider Punktwolken

Es werden verschiedene, bewährte ICP-Varianten auf mehrere Punktwolken angewendet und deren Überdeckung anschließend anhand der bereits erwähnten Metrik verglichen. Es soll festgestellt werden, ob sich eine angemessene Qualität durch den

gewählten Ansatz realisieren lässt und welche weiteren Optimierungspotentiale für den Testaufbau bestehen.

### 3) Prototyp-Applikation

Nachdem der Algorithmus die erforderlichen Koordinatentransformationen berechnet hat, werden letztere auf 3D-Livedaten angewendet. Die endgültige Implementierung soll die Arbeitsumgebung in einem einzigen Videostrom – bestehend aus zwei zusammengeführten 3D-Punktwolken-Livestreams – in hoher Qualität darstellen.

## 1.3 Aufbau der Arbeit

Zu Beginn werden in Kapitel 2 die notwendigen theoretischen Grundlagen und verwendeten Algorithmen erläutert. Des Weiteren werden die verwendeten Methoden zur Vorverarbeitung der Punktwolken beschrieben. Anschließend wird in Kapitel 3 die ausgewählte Implementierung anhand mehrerer Datensätze aus verschiedenen Kollaborationsszenarien getestet und verglichen.

Die Ergebnisse werden in weiterer Folge anhand der beiden Kriterien Ressourcenverbrauch und Genauigkeit bewertet. Zuletzt endet diese Arbeit unter Kapitel mit einer Zusammenfassung sowie einem kurzen Ausblick.

## 2 Theoretische Grundlagen

In diesem Kapitel werden die notwendigen Grundlagen erläutert. In Abschnitt 2.1 wird zunächst ein Überblick des aktuellen Standes der Forschung gegeben. Nach einer Beschreibung des Kameraaufbaus in Abschnitt 2.2 werden anschließend in Abschnitt 2.3 die theoretischen Grundlagen erläutert. Diese sind notwendig für die weitere Beschreibung des ICP in Abschnitt 2.4 und dessen Weiterentwicklungen im Abschnitt 2.5. In Abschnitt 2.6 wird die Reihenfolge der Vorverarbeitungen innerhalb einer Punktwolken-Registrierung veranschaulicht. Im Anschluss an die allgemeine Beschreibung von Vorverarbeitungsmethoden und deren Auswirkungen werden im Abschnitt 2.6.2 angewendete Vorverarbeitungsschritte beschrieben.

### 2.1 Stand der Forschung

In der maschinellen Bildverarbeitung gibt es Situationen, in welchen mehrere ähnliche Bilder übereinandergelegt werden müssen. Beispielsweise muss es autonomen Robotern möglich sein, ihre Umgebung zu erkennen. Gleichzeitig soll eine interne Umgebungskarte zur Navigation aufgebaut werden. Aktuelle 3D-Bilder eines oder mehrerer Sensoren werden hierzu ständig erfasst und gespeichert. Bewegt sich der Roboter, besitzen diese Bilder im Vergleich zu vorigen Aufnahmen unterschiedliche Blickwinkel und Positionen. Beide Bilder müssen so lange rotiert, verschoben und skaliert werden, bis eine bestmögliche Überlagerung erreicht wird [1]. Für diese Koordinatentransformation wird ein effizienter Algorithmus benötigt.

3D-Bilder können auch als 3D-Punktwolken bezeichnet werden [2]. Sie stellen eine Menge von Bildpunkten im dreidimensionalen Raum dar. Die Position jedes dieser Punkte wird durch X-, Y- und Z-Koordinaten beschrieben:  $P = \begin{bmatrix} x & y & z \end{bmatrix}^T, x, y, z \in \mathbb{R}$

Eine gängige Metrik zur Bewertung der Überlagerung von Punktwolken ist die euklidische Distanz zwischen den korrespondierenden Punkten beider Datensätze. Durch schrittweise Minimierung einer definierten Fehlerfunktion (siehe Gleichung 9) mittels Transformation der Punktwolken kann diese Deckung erreicht werden. Mehrere ähnliche Bilder computergestützt zusammenzuführen, wird als Bildregistrierung bezeichnet.

Besl und McKay [1] stellen mit ihrem ICP eine effiziente Lösung für das Problem der Überlappung von zwei Punktwolken vor. Dieser nähert sich durch schrittweise, sich wiederholende Berechnungen, einer optimalen Überlappung beider Punktwolken.

Unterscheiden sich Aufnahmepositionen und -winkel der 3D-Bilder stark voneinander, erzielt der ICP keine zuverlässigen Ergebnisse [3]. Durch eine manuelle (grobe) Vorausrichtung beider Punktwolken vor Anwendung des ICP lässt sich ein besseres Endergebnis er-

reichen [4]. Diesen Ansatz verfolgt die Arbeit von Zhu et al. [3]. Die Autoren schlagen eine Methode für die Lösung dieses Problems vor, indem sie mittels sogenanntem Constrained-ICP (C-ICP) in erster Instanz eine grobe Ausrichtung der Punktwolken durchführen. Dies ist möglich, da die Abstände und Winkel von statisch montierten Kameras bekannt sind. Des Weiteren berücksichtigt der C-ICP ausschließlich Punkte, welche sich in der Aufnahme beider Kameras befinden.

Bellekens et al. [4] vergleichen sechs verschiedene und häufig verwendete ICP-Varianten für die 3D-Registrierung. Dazu wird ein Benchmark definiert: Die Präzision wird anhand der mittleren euklidischen Distanz, die Leistung anhand der Iterationen und die Robustheit durch Tests an verschiedenen 3D-Punktwolken ermittelt. Die ICP-Varianten Singular-Value-Decomposition ICP (SVD-ICP), Point-To-Surface ICP (ICP-PTS), Generalized-ICP (G-ICP) sowie der Nonlinear-ICP (NL-ICP) erreichten hierbei die besten Resultate. Die stetige Weiterentwicklung des ICP bringt ständige, nützliche Verbesserungen mit sich.

## 2.2 Kameraaufbau

Wie in Abschnitt 1.1 beschrieben, wird für die Erfassung aller Bereiche eines Mensch-Roboter-Arbeitsplatzes ein Aufbau mit mehreren Kameras benötigt (siehe Abbildung 1).

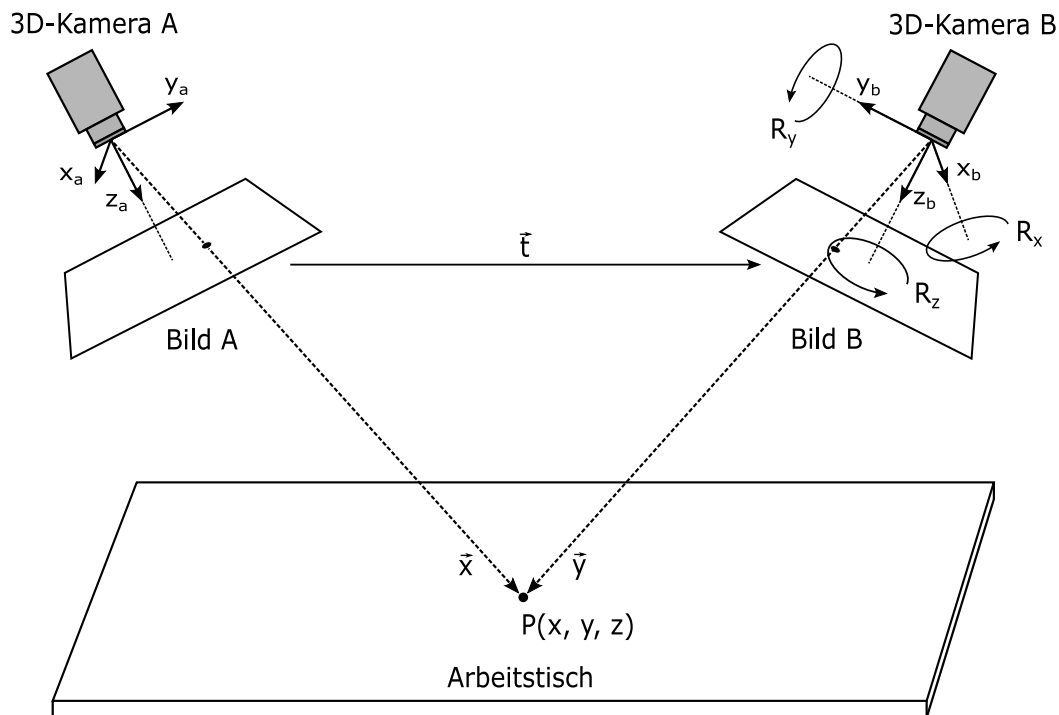


Abbildung 1: Kameraaufbau

Beide Kameras besitzen ein eigenes 3D-Koordinatensystem, bestehend aus den Dimensionen  $X$ ,  $Y$  und  $Z$ . Sei  $P$  ein gegebener Punkt mit den Koordinaten  $x$ ,  $y$  und  $z$ . Die

Koordinaten von  $P$  unterscheiden sich jedoch Bild A und Bild B. Die Werte von insgesamt 6 Freiheitsgraden sind veränderbar. Drei Freiheitsgrade ergeben sich durch eine Verschiebung  $\vec{t}$  in den X-, Y-, Z-Achsen und weitere drei durch Rotationen  $R_x, R_y, R_z$  derselben Achsen. Ein Punkt  $P$  in einem 3D-Koordinatensystem kann als dreidimensionaler Vektor beschrieben werden. Aus Sicht von Kamera A soll dieser Vektor zunächst als  $\vec{x}$  und aus Sicht von Kamera B als  $\vec{y}$  bezeichnet werden. Ziel ist es, das Bild von Kamera B so zu verschieben und zu drehen, dass es mit dem Bild von Kamera A übereinstimmt. Das bedeutet, dass  $\vec{x}$  dieselbe Länge und Richtung wie  $\vec{y}$  haben muss. Dieser Vorgang lässt sich über Koordinatentransformationen beschreiben [4].

## 2.3 Koordinatentransformationen

Der Zusammenhang von  $\vec{x}$  und  $\vec{y}$  kann über eine Transformationsmatrix  $T$ , bestehend aus einer Translation  $\vec{t}$  und einer Rotation  $R$ , dargestellt werden [4].

Die Rotation in drei Freiheitsgraden lässt sich wie folgt abbilden:

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \{r_{11}, r_{12}, \dots, r_{33}\} \subset \mathbb{R} \quad (1)$$

Eine Translation in drei Freiheitsgraden ist mit einem Vektor  $\vec{t}$  darstellbar:

$$\vec{t} = \begin{bmatrix} t_1 & t_2 & t_3 \end{bmatrix}^T, t_1, t_2, t_3 \in \mathbb{R} \quad (2)$$

$R$  und  $\vec{t}$  sollen in einer Transformationsmatrix zusammengefasst werden, sodass eine Multiplikation eines Punktes  $P$  mit  $T$  einen um  $\vec{t}$  verschobenen Punkt  $P'$  ergibt.

Sei  $P = \begin{bmatrix} 0 & 0 & 0 \end{bmatrix}^T$  der Ausgangspunkt und  $I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$  die Einheitsmatrix im  $\mathbb{R}^3$ .

$P$  soll durch  $\vec{t} = \begin{bmatrix} 3 & 5 & 8 \end{bmatrix}^T$  um +3 in der X-Koordinate, um +5 in der Y-Koordinate und um +8 in der Z-Koordinate verschoben werden, sodass  $P' = \begin{bmatrix} 3 & 5 & 8 \end{bmatrix}^T$ .

Durch das Hinzufügen einer zusätzlichen Dimension werden die Koordinaten zu sogenannten homogenen Koordinaten und eine Verschiebung des Ausgangspunktes um Absolutwerte durch eine Transformationsmatrix wird ermöglicht. Die Gleichung (3) zeigt die Translation des Punktes mittels homogener Koordinaten und  $P'$  als korrekt verschobenen Punkt.

$$P' = \begin{bmatrix} 0+3 \\ 0+5 \\ 0+8 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 5 \\ 0 & 0 & 1 & 8 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (3)$$

Zuletzt muss die Einheitsmatrix aus  $T$  lediglich durch die gewünschte Rotationsmatrix  $R$  ersetzt werden. Es ergibt sich eine Transformationsmatrix, welche eine Translation und Rotation vereinigt:

$$T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

Diese Transformationen können auf die gesamten Punkte einer Punktwolke angewendet werden. Hierzu muss jeder Punkt bzw. Vektor mit  $T$  multipliziert werden.

## 2.4 Grundlegende Vorgehensweise des ICP

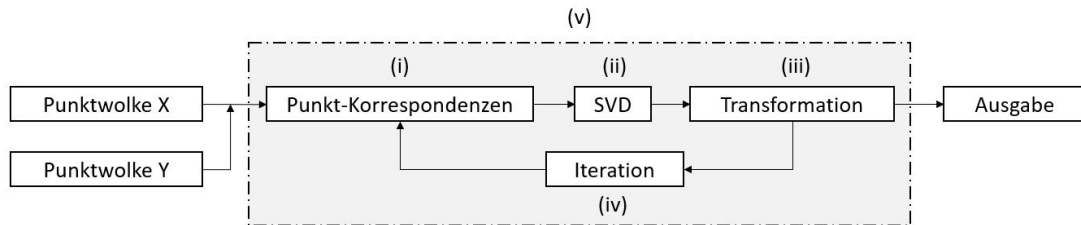


Abbildung 2: Übersicht der einzelnen Schritte des ICP, adaptiert aus [4]

Für zwei Punktwolken,  $X$  und  $Y$ , bestehend aus  $N_x$  respektive  $N_y$  Punkten soll eine Möglichkeit gefunden werden, sie übereinander zu legen. Dazu wird eine Punktwolke so gedreht und verschoben, bis sich beide Punktwolken so gut wie möglich überlagern

$$X = \{x_i \in \mathbb{R}^3 \mid i = 1, \dots, N_x\} \quad (5)$$

$$Y = \{y_i \in \mathbb{R}^3 \mid i = 1, \dots, N_y\} \quad (6)$$

Der ICP verfolgt einen iterativen Ansatz und besteht aus folgenden Komponenten [5]:

- (i) Bestimmung der korrespondierenden Punktpaare zwischen  $X$  und  $Y$ .
- (ii) Berechnen einer Transformation  $T_j$ , um  $Y$  an  $X$  ein Stück weiter anzunähern.
- (iii) Anwenden der aktuell geschätzten Transformation  $T_j$  auf  $Y$ .

- (iv) Minimierung der Fehlerfunktion  $E(T_j)$ .
- (v) Weiterberechnung von  $T$  durch Multiplikation mit  $T_j$
- (vi) Iterationsprozess bis  $E(T)$  unter einen bestimmten Schwellwert gelangt und weitere Iterationen keine Verbesserung bringen.

$T$  wird bei jeder Iteration mit dem aktuellen  $T_j$  multipliziert. Das bedeutet, dass  $T$  sozusagen stückweise die notwendige Gesamttransformation von  $Y$  zu  $X$  abbildet. Die aufgezählten Schritte werden in den folgenden Abschnitten erläutert.

### 2.4.1 Bestimmung der Punkt-Korrespondenzen

Zunächst wird in der ersten Punktwolke eine bestimmte Anzahl zufälliger Punkte  $\{x_1, \dots, x_i\} \subset X$  ausgewählt. Hierzu werden zuerst die Distanzen  $D = \{d_i, i = 1, \dots, N_x \mid d_i = (x_i - y_i)^2\}$  aller Punkt-Korrespondenzen berechnet. Anschließend wird der Median  $\tilde{d}$  daraus ermittelt und ein bestimmter Faktor  $k$  definiert. Dieser Faktor kann empirisch ermittelt werden und bestimmt die Anzahl der verwendeten Punkt-Korrespondenzen, denn alle Punkt-Korrespondenzen, welche eine größere Distanz als  $\tilde{d} \cdot k$  zueinander besitzen, werden aus den bestehenden Korrespondenzen ausgeschlossen:

$$d_i > k \cdot \tilde{d} \tag{7}$$

Weiterführend werden ihnen ihre jeweils nächsten Punkte  $y_i \in Y$  aus der zweiten Punktwolke anhand ihrer euklidischen Distanz zugeordnet (Abbildung 3 ).

Bei jeder Iteration  $j$  werden weitere Punkt-Korrespondenzen ausgeschlossen. Da sich die Punktwolken schrittweise einander annähern, muss der Faktor  $k$  bei jedem Schritt kleiner werden. Andernfalls berücksichtigte dies wieder mehr Punkt-Korrespondenzen. Dieser Faktor kann je nach ICP-Variante verschieden sein.

### 2.4.2 Minimierung der Fehlerfunktion

Pro Iteration  $j$  wird eine Matrix  $T_j$  neu berechnet, auf die Punktwolke  $Y$  angewendet und anschließend auf die Matrix  $T$  angewendet. Nach Beendigung des Prozesses ergibt sich  $T$  sozusagen aus Multiplikation bzw. Hintereinanderausführung aller einzelnen  $T_j$ . Die Punkte aus  $X$  und  $Y$  stehen über eine bestimmte Transformationsmatrix in Relation und  $T$  soll die Unterschiede zwischen beiden Punktwolken ausgleichen. Gleichung 8 beschreibt diesen Vorgang. Da die Punktwolken sich nur auf ein bestimmtes Minimum annähern, ist es unwahrscheinlich, dass alle Punkte perfekt überlagert werden.



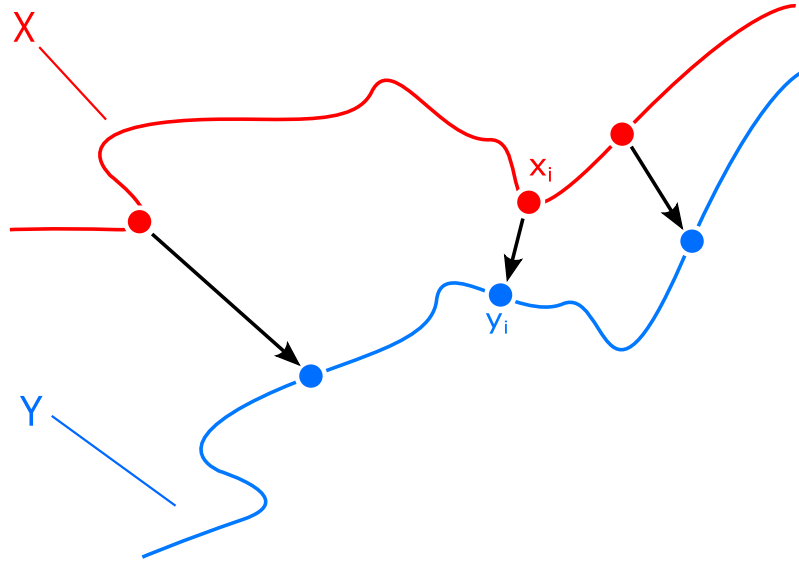


Abbildung 3: Wahl der Punkt-Korrespondenzen über die nächsten Punkte, adaptiert aus [4]

$$\{\forall x_i \in X \mid x_i \approx T \cdot y_i\} \quad (8)$$

Generell lässt sich das iterative Angleichen beider Punktwolken als ein Optimierungsproblem darstellen. Das bedeutet, dass versucht wird, eine Fehlerfunktion zu minimieren bzw. einem Optimum in einem bestimmten Bereich anzunähern. Das globale Optimum wäre die vollständige Überlappung beider Punktwolken. Je kleiner das Ergebnis der Fehlerfunktion, desto besser die Überlappung.

Die Fehlerfunktion ist wie folgt definiert, und ihre Herleitung ist in [6, Abschnitt 3.1.1] zu finden:

$$E(T_j) = \frac{1}{N_y} \sum_{i=1}^{N_y} (x_i - R \cdot y_i - \vec{t})^2 \quad (9)$$

Da  $Y$  transformiert werden soll, wird  $E(T_j)$  über alle Punkte von  $Y$  (siehe Gleichung 6) ausgeführt. Ähnlich dem Gauß-Newton-Verfahren wird ein Minimum der Fehlerquadratsumme gesucht [7]. Der durchschnittliche quadratische Fehler  $E(T_j)$  nach einer Iteration (Rotation und Translation) ergibt sich aus der Berechnung der Summe der quadratischen, euklidischen Distanzen der einzelnen korrespondierenden Punkte  $x_i$  und  $y_i$ .  $E(T)$  ist ein Maß für die Überlappung beider Punktwolken. Das globale Minimum ist hierbei als das optimale Ergebnis zu sehen.

Des Weiteren besteht eine hohe Wahrscheinlichkeit, sich mittels ICP nur an ein lokales Minimum anzunähern [7]. Dies lässt sich dadurch begründen, dass ab einer bestimm-

ten Anzahl von Iterationen, nach einer weiteren Transformation, keine Verbesserung der Überlappung mehr auftritt. Tritt der Fall ein, dass die Überlappung nicht zufriedenstellend ist, und keine weitere Verbesserung mehr möglich ist, wurde ein lokales Minimum erreicht. Hierbei ist beispielsweise eine andere Methode zur Pre-Transformation [3] oder die Verwendung einer anderen ICP-Variante notwendig [4].

### 2.4.3 Singular-Value-Decomposition

Um die beiden Punktwolken einander anzugleichen, muss die Fehlerfunktion  $E(T)$  minimiert werden. Arun, Huang und Blostein stellten hierzu eine Methode basierend auf der sogenannten Singular-Value-Decomposition (SVD) vor [8]. Durch diese Methode kann die Transformation und Ausrichtung zwischen zwei Punktwolken geschätzt werden [9], [10]. Hierzu werden jedoch die optimalen Punkt-Korrespondenzen für jedes Punktpaar  $x_i, y_i$  benötigt, welche üblicherweise nicht verfügbar sind.

Die Transformation einer Iteration  $T_j$  wird in diesem Abschnitt als  $R$  und  $\vec{t}$  getrennt behandelt. Bei jeder Iteration wird die SVD mit anderen Punkt-Korrespondenzen wiederholt. Somit gleichen sich die Punktwolken schrittweise einander an.

#### Vorbereitung der Punktwolken für die SVD

Zu Beginn des ICP erfolgt eine Pre-Transformation  $T_{\text{CoM}}$  von  $X$  und  $Y$ , sodass ihre Massenmittelpunkte (Center of Mass) übereinstimmen. Dies zeigt eine Verringerung der benötigten Iterationen bis zur Überlappung um 2 bis 5 [1]. Die Massenmittelpunkte von  $X$  und  $Y$  werden  $\mu_x$  respektive  $\mu_y$  genannt und sind wie folgt definiert:

$$\mu_x = \frac{1}{N_x} \sum_{i=1}^{N_x} x_i \quad (10)$$

$$\mu_y = \frac{1}{N_y} \sum_{i=1}^{N_y} y_i \quad (11)$$

Durch Subtraktion der jeweiligen Massenmittelpunkte von allen Punkten der betreffenden Datensätze ergeben sich zwei neue Punktwolken:

$$X' = \{x'_1, \dots, x'_{N_x} \mid x'_i = x_i - \mu_x\} \quad (12)$$

$$Y' = \{y'_1, \dots, y'_{N_y} \mid y'_i = y_i - \mu_y\} \quad (13)$$

#### Berechnung der SVD

Laut [8] werden zunächst alle Punkt-Korrespondenzen  $x'_i, y'_i$ , welche in der aktuellen Iteration berücksichtigt werden, als Vektoren miteinander multipliziert und die jeweils entstehenden 3 x 3 Matrizen aufsummiert. Die dadurch entstehende Korrelationsmatrix  $H$

enthält Informationen über die Streuung und Korrelationen zwischen den einzelnen Vektoren bzw Punkt-Korrespondenzen. Nuchter [6] beschreibt die Herleitung von  $H$ .

$$H = \sum_{i=1}^{N_x} x'_i y_i'^T = \sum_{i=1}^{N_x} \begin{bmatrix} x'_{i1} \\ x'_{i2} \\ x'_{i3} \end{bmatrix} \cdot \begin{bmatrix} y'_{i1} \\ y'_{i2} \\ y'_{i3} \end{bmatrix}^T = \sum_{i=1}^{N_x} \begin{bmatrix} x'_{i1}y'_{i1} & x'_{i1}y'_{i2} & x'_{i1}y'_{i3} \\ x'_{i2}y'_{i1} & x'_{i2}y'_{i2} & x'_{i2}y'_{i3} \\ x'_{i3}y'_{i1} & x'_{i3}y'_{i2} & x'_{i3}y'_{i3} \end{bmatrix} \quad (14)$$

Der beschriebene Ansatz sieht nun eine Singulärwertzerlegung von  $H$  vor, da hierdurch die für die Berechnung der Rotation  $R$  benötigten Matrizen  $U$  und  $V$  ermittelt werden:

$$H = U \Sigma V^T \quad (15)$$

Mit  $U$  und  $V$  ist eine innerhalb der aktuellen Iteration ausgleichende Rotation berechenbar:

$$R = UV^T \quad (16)$$

Nach angewandeter Rotation  $R$  ergibt sich die Translation  $\vec{t}$  durch einfaches Subtrahieren der Mittelpunkte beider Punktwolken:

$$\vec{t} = \mu_x - R\mu_y \quad (17)$$

Mittels  $\vec{t}$  und  $R$  kann nun die Fehlerfunktion (siehe Gleichung 9) minimiert werden und die Punktwolken ein Stück weiter angenähert werden. Hierbei handelt es sich lediglich um eine Transformation anhand der aktuellen Punkt-Korrespondenzen. Weitere Iterationen mit neuen Korrespondenzen sind notwendig, um die beiden 3D-Punktwolken bis ans mögliche Optimum anzunähern. Eine Verbesserung ist allerdings nicht garantiert, und der Algorithmus beendet seine Iterationen sofern dieser Fall eintritt.

## 2.5 Variationen des ICP

Um für spezifische Anforderungen bessere Ergebnisse zu erzielen, existieren diverse Abwandlungen des ICP. In diesem Abschnitt werden die für diese Arbeit relevanten Varianten erläutert.

### 2.5.1 Nonlinear-ICP

Der im Abschnitt 2.4 beschriebene Standard-ICP verwendet als Kostenfunktion den quadratischen verbleibenden Abstand zwischen Punkt-Korrespondenzen. Das bedeutet, dass Punktpaare mit großen Abständen das Ergebnis besonders stark beeinflussen. Der NL-ICP hingegen verwendet zur Fehlerminimierung den Levenberg-Marquard Algorithmus, welcher die Verwendung von Kostenfunktionen erlaubt, die stabiler gegenüber Ausreißern sind [11].

Eine Möglichkeit ist die Verwendung des Absolutwertes der Distanz als Kostenfunktion. Eine Absolutwertfunktion ist allerdings im Ursprung nicht ableitbar, was das Finden einer optimalen Lösung bei der Minimierung schwierig macht. Als Kompromiss kann beispielsweise die in Abbildung 4 dargestellte Huber-Verlustfunktion [12] als stabile Kostenfunktion für den NL-ICP verwendet werden. Sie verwendet eine quadratische Funktion um den Ursprung und geht dann in eine lineare Funktion über [4], [11], [13].

$$e^2(x) = \begin{cases} \frac{1}{2}x^2 & \text{für } |x| < k \\ k|x| - \frac{k^2}{2} & \text{für } |x| \geq k \end{cases} \quad (18)$$

Dabei ist  $x$  der verbleibende Abstand und  $k$  ein empirisch festgestellter Schwellenwert. Im Gegensatz zum Standard-ICP ist die Berechnung pro Iteration aufwändiger, dafür konvergiert der NL-ICP im Regelfall schneller, wodurch die gesamte Berechnung nicht länger dauert [11].

### 2.5.2 Generalized-ICP

Der Standard-ICP versucht die euklidische Distanz von Punkt-Korrespondenzen direkt zu minimieren. Es wird dabei von einer Punkt-zu-Punkt-Korrespondenz gesprochen. Um damit ein ideales Ergebnis zu erhalten, benötigt man Daten, bei denen die Punkte in den zu registrierenden Punktwolken exakte Übereinstimmungen haben. Das ist allerdings nur der Fall, wenn die Punktwolken entweder generiert wurden oder Teilauszüge aus einer Quellpunktwolke sind. In praktischen Anwendungsfällen, bei denen zwei oder mehrere Aufzeichnungen in Relation gesetzt werden sollen, ist das in der Regel nicht der Fall. Es werden vom Sensor durch die Verschiebung der Aufnahmeposition und Diskretisierung der

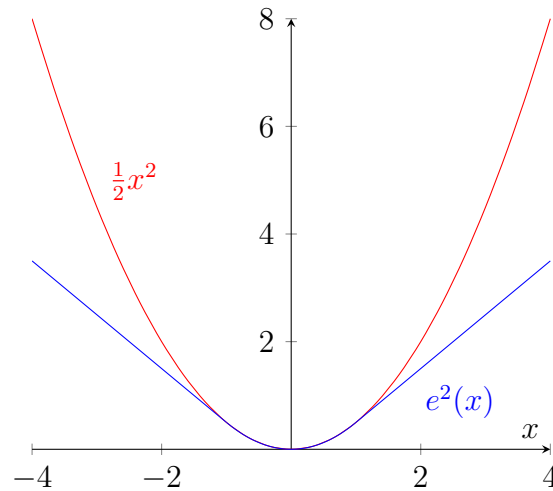


Abbildung 4: Die blaue Linie zeigt eine Huber-Verlustfunktion mit  $k = 1$  (siehe Gleichung 18). Die rote Linie beschreibt den quadratischen Teil der Verlustfunktion.

aufgenommen Werte Punkte an leicht unterschiedlichen Positionen des gleichen Objekts aufgenommen. Bei der Registrierung mittels Standard-ICP entsteht dadurch ein Fehler, der beispielsweise wie in Abbildung 5 aussehen kann. Es ist dort ersichtlich, dass die blaue Punktwolke eigentlich nach links verschoben sein sollte, um eine genaue Überlappung der Struktur zu erreichen. Diese Verschiebung führt allerdings zu einem größeren Durchschnittsabstand der Punkte und wird daher mit einer Punkt-zu-Punkt Metrik nicht vorgenommen [14], [15].

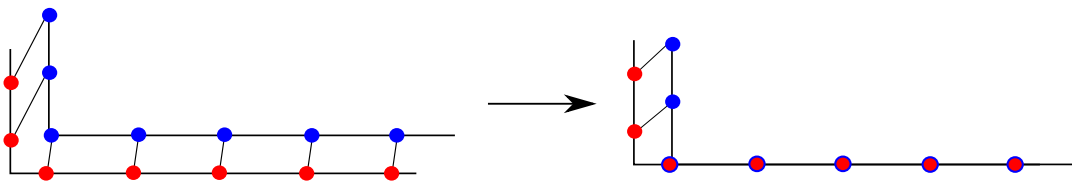


Abbildung 5: Diese Abbildung zeigt ein Objekt, wie es in zwei unterschiedlichen Punktwolken (rot bzw. blau) aufgenommen wurde. Die linke Seite zeigt die gefundenen Punkt-Korrespondenzen, die rechte Seite das Ergebnis einer Punkt-zu-Punkt Registrierung (adaptiert aus [15]).

Der G-ICP wird in [14] als Lösungsvorschlag für dieses Problem vorgestellt. Er trifft die Annahme, dass die meisten aufgezeichneten Daten aus Punkten bestehen, die lokale Flächen mit den Punkten in ihrer direkten Umgebung bilden. Der G-ICP minimiert den Abstand dieser lokalen Flächen (Fläche-zu-Fläche-Registrierung).

In der Implementierung folgt der G-ICP bis auf den Schritt der Minimierung der Fehlerfunktion dem im Abschnitt 2.4 beschriebenen Ablauf. Der G-ICP baut dabei auf ein Wahrscheinlichkeitsmodell, bei dem die Existenz der Punktmengen  $\hat{A} = \{\hat{a}_i\}$  und  $\hat{B} = \{\hat{b}_i\}$  angenommen wird. Diese Punktwolken beschreiben die tatsächlichen Punkte im Raum,

während die gemessenen Punktwolken  $A = \{a_i\}$  und  $B = \{b_i\}$  mittels mehrdimensionaler Normalverteilung aus den tatsächlichen generiert werden.  $C_i^A$  und  $C_i^B$  sind die Kovarianz-Matrizen für die Punkte am zugehörigen Index.

$$a_i \sim \mathcal{N}(\hat{a}_i, C_i^A) \quad \text{und} \quad b_i \sim \mathcal{N}(\hat{b}_i, C_i^B) \quad (19)$$

Gleichung 20 wird im G-ICP für die Bestimmung der Transformation mit dem niedrigsten Fehler verwendet, wobei die Definition  $d_i(\mathbf{T}) = b_i - \mathbf{T}a_i$  gilt und  $\mathbf{T}$  die Transformation von  $A$  nach  $B$  ist. Es wird angenommen, dass alle Punkte, für die keine Korrespondenzen gefunden wurden, bereits aus  $A$  und  $B$  entfernt wurden [14].

$$\mathbf{T}_{min} = \arg \min_{\mathbf{T} \in \mathbb{R}^{4 \times 4}} \sum_i d_i(\mathbf{T})^T (C_i^B + \mathbf{T}C_i^A \mathbf{T}^T)^{-1} d_i(\mathbf{T}) \quad (20)$$

Die Kovarianz-Matrizen  $C_i^A$  und  $C_i^B$  werden mit einer hohen Kovarianz entlang der lokalen Fläche und einer niedrigen Kovarianz normal zur Fläche modelliert. Daraus ergibt sich, dass der Registrierungsfehler für Punkte die nebeneinander auf derselben Fläche liegen, niedriger ist als für Punkte, die den gleichen Abstand normal zu ihren lokalen Flächen haben. Es wird also die Struktur der aufgenommenen Objekte in der Fehlerfunktion berücksichtigt. Abbildung 6 zeigt eine schematische 2D-Darstellung einer G-ICP Registrierung. Für die einzelnen Punkte sind Ellipsen dargestellt, in denen die Positionen der korrespondierenden Punkte mit einer hohen Wahrscheinlichkeit erwartet wird. Die Form der Ellipsen wird dabei durch die Kovarianz-Matrizen bestimmt [14].

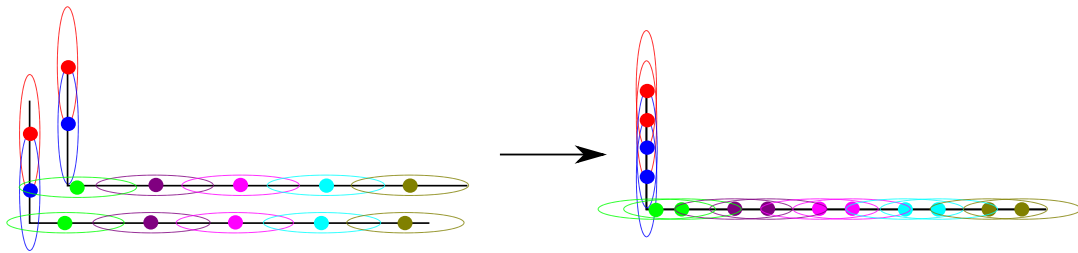


Abbildung 6: Schematische Darstellung einer Registrierung mit G-ICP. Die linke Seite zeigt zwei Punktwolken, die Aufnahmen des gleichen Objekts sind, wobei gleiche Farben Punkt-Korrespondenzen kennzeichnen. Die rechte Seite zeigt das Ergebnis der Registrierung (adaptiert aus [14]).

Durch eine Wahl der Einheitsmatrix für  $C_i^B$  und der Nullmatrix für  $C_i^A$  kann gezeigt werden, dass der Standard-ICP nur ein Sonderfall des G-ICP ist, da sich dadurch für die Fehlerfunktion wieder die Fehlerquadratsumme ergibt [14].

## 2.6 Vorverarbeitung von Punktwolken

Als Vorverarbeitung werden Arbeitsschritte bezeichnet, welche vor der Hauptanwendung an den erzeugten Punktwolken durchgeführt werden. Durch die Vorverarbeitung werden die Punktwolken für die spezifische Hauptanwendung optimiert [16].

Nach [17] ist in Abbildung 7 eine typische Abfolge von Arbeitsschritten für die Punktwolken-Registrierung dargestellt. Im Arbeitsschritt „Punktwolken Generierung“ werden mit zwei 3D-Kameras die Punktwolken erzeugt. Anschließend erfolgt das „Vorverarbeitung“ beider Punktwolken. In diesem Projekt wird nach der Vorverarbeitung die „Punktwolken-Registrierung“ durchgeführt.



Abbildung 7: Grobgranulare Betrachtung der unterschiedlichen Arbeitsschritte einer Punktwolken-Registrierung.

### 2.6.1 Gründe für die Vorverarbeitung

Die Rechendauer und Korrektheit des Ergebnisses einer Punktwolken-Registrierung ist abhängig von der Qualität der Eingabe-Punktwolken. Fehler in der erzeugten Punktwolke können eine erhöhte Rechendauer oder eine falsche Punktwolken-Registrierung zur Folge haben. Die Rechendauer der Punktwolken-Registrierung wird ebenfalls von der Anzahl an Punkten einer Punktwolke beeinflusst. Eingabe-Punktwolken können Punkte aufweisen, die für die Hauptanwendung keine oder vernachlässigbar geringe Informationen zur Verfügung stellen. Eine hohe Rechendauer oder fehlerhafte Berechnungen bei der Punktwolken-Registrierung können zu unbrauchbaren Ergebnissen führen [18].

Zwei in der Literatur für das Auftreten von verzichtbaren Punkten innerhalb der erzeugten Punktwolken als verantwortlich angesehene Ursachen sind folgende [16]:

- Die Sensoren erfassen einen größeren Bereich im Raum als den relevanten Arbeitsbereich. Die erzeugten Punktwolken beinhalten Informationen, die für die darauffolgenden Arbeitsschritte nicht relevant sind. Diese Situation ergibt sich unter anderem durch die Montage der Kameras an der Decke und deren festen Blickwinkel.
- Der Detailgrad und die Auflösung der erzeugten Punktwolken sowie die daraus resultierende hohe Anzahl an einzelnen Punkten ist für die weitere Verarbeitung nicht

notwendig. Relevante Objekte können mit einer geringeren Anzahl an Punkten sinnvoll dargestellt werden.

Die Ursachen von Verunreinigung in den Punktwolken sind einerseits aufgenommenes Sensorrauschen bei der Erzeugung der Punktwolken und andererseits einzelne Punkte, die als Ausreißer in der Punktwolke definiert werden können [19].

In beiden Fällen der Verunreinigung handelt es sich um fehlerhafte Punkte. Kostengünstige Sensoren weisen vermehrt Verunreinigungen auf. Verunreinigungen treten ebenfalls bei Objekten auf, deren Oberflächenstrukturen stark spiegeln und somit nicht optimal vom Sensor erfasst werden können [18]. Aufgrund des Aufnahmeverfahrens der Sensoren können sich Wärmequellen innerhalb des Aufnahmebereichs genauso wie externe und nicht gleichmäßige Lichtquellen ungünstig auf die Erzeugung von Punktwolken auswirken. Manuell einstellbare Kalibrierparameter wirken sich ebenfalls auf die Qualität der erzeugten Punktwolken aus.

Folgende Vorverarbeitungsmethoden werden angewandt, um eine verbesserte Eingabepunktwolke für die Punktwolken-Registrierung zur Verfügung zu stellen.

**1) Das Begrenzen der Punktwolke (pass through)**

Dargestellte Bereiche in den Punktwolken, die keine Relevanz für die Punktwolken-Registrierung haben werden aus den Punktwolken entfernt. Dadurch wird eine Verkürzung der Rechenzeit bei weiteren Arbeitsschritten erreicht und unnötige Störeinflüsse werden frühzeitig entfernt [16].

**2) Dezimierung der Punkte (downsampling)**

Mit keinem oder vertretbar geringem Informationsverlust werden einzelne Punkte einer Punktwolke entfernt. Durch die geringere Anzahl an Punkten wird eine schnellere Verarbeitung in der Hauptanwendung ermöglicht und eine Glättung des Bildes erreicht [16].

**3) Rausch- und Ausreißer-Entfernung (noise and outlier removal)**

Fehler in Form von Ausreißern und Sensorrauschen in den erzeugten Punktwolken werden entfernt [19]. Diese Vorverarbeitung erhöht die Qualität der Punktwolken, ohne dabei relevante Informationen zu verlieren. Die höhere Qualität der Punktwolke reduziert die Rechendauer und die fehlerhaften Berechnungen der Punktwolken-Registrierung.

**4) Bewegung der kleinsten Quadrate, MLS (moving least squares)**

Dieser Vorverarbeitungsschritt wird zur Flächenrekonstruktion von Objekten der erzeugten Punktwolken verwendet [20]. Eine ebene Tischplatte die in der erzeugten



Punktwolke als wellige Tischplatte dargestellt wird, kann mittels MLS Vorverarbeitungsschritt auf die erzeugte Punktwolke wieder als ebene Tischplatte dargestellt werden.

### 2.6.2 Angewandte Verfahren der Vorverarbeitung

In diesem Projekt werden vier eigenständige Verfahren für die Vorverarbeitung der Punktwolken verwendet. Sie decken jeweils eine im vorangegangenen Abschnitt erläuterte Vorverarbeitungsmethode ab. Dieser Abschnitt erläutert die Aufgaben und Eigenschaften der einzelnen Verfahren. Die Reihenfolge und Namen der einzelnen Verfahren sind in Abbildung 8 dargestellt.

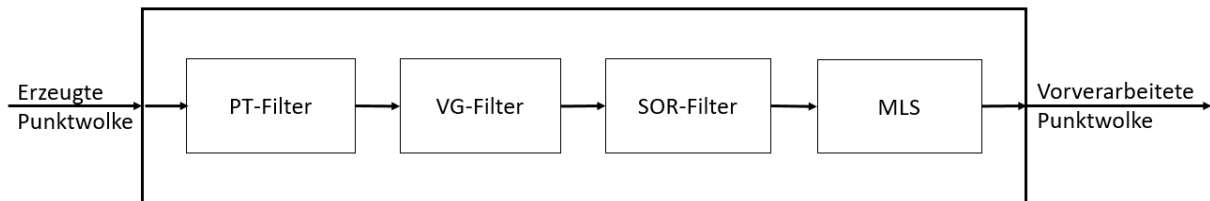


Abbildung 8: Die vier unabhängigen Verfahren, die in Summe die gesamte Vorverarbeitung bilden.

#### Pass Through Filter

Die Aufgabe des Pass Through Filter (PT-Filter), zu Deutsch Durchgangsfiler, ist wie in Abschnitt 2.6.1 beschrieben, die Beschneidung der Punktwolken. Dem Durchgangsfiler wird ein Minimum- und Maximum-Grenzwert für X-, Y- und Z-Achsen parametrisiert.

Die Koordinaten jedes Punktes einer Punktwolke werden mit den parametrisierten Grenzwerten verglichen. Punkte, die sich innerhalb aller definierten Grenzwerte befinden, verbleiben in der Punktwolke, alle anderen Punkte werden vom Durchgangsfiler aus der Punktwolke entfernt [16]. Wird eine Achse für den Durchgangsfiler nicht parametrisiert, ist diese Achse für den Vergleich nicht relevant.

Die Funktionsweise des Durchgangsfilters wird im folgenden theoretischen Beispiel veranschaulicht. Für das Beispiel wird eine Punktwolke mit fünf Punkten im zweidimensionalen Raum verwendet. Die Eingabepunkte  $P_1$  bis  $P_5$  sind in Abbildung 9 in rot und grün dargestellt. Die Punkte besitzen die Koordinaten:

$$P_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, P_2 = \begin{pmatrix} 2 \\ 2 \end{pmatrix}, P_3 = \begin{pmatrix} -1 \\ -1 \end{pmatrix}, P_4 = \begin{pmatrix} -3 \\ 3 \end{pmatrix}, P_5 = \begin{pmatrix} -1 \\ 1 \end{pmatrix} \quad (21)$$

Die Grenzwerte des Durchgangsfilters für die x Achse wurden auf  $X_{min} = 0$  und  $X_{max} = 4$  parametrisiert. Der Durchgangsfiler vergleicht die X-Koordinaten aller Punkte. Ist die X-

Koordinate eines Punktes kleiner als der parametrisierte  $X_{min}$  oder größer als der parametrisierte  $X_{max}$  wird dieser Punkt aus der Punktwolke entfernt.  $P_3$  mit  $x = -1$ ,  $P_4$  mit  $x = -3$  und  $P_5$  mit  $x = -1$  sind kleiner als  $X_{min}$  des Durchgangsfilters und werden aus der Punktwolke entfernt. Die Punktwolke nach der Filterung ist in Abbildung 9 durch die grünen Punkte  $P_1$  und  $P_2$  dargestellt.

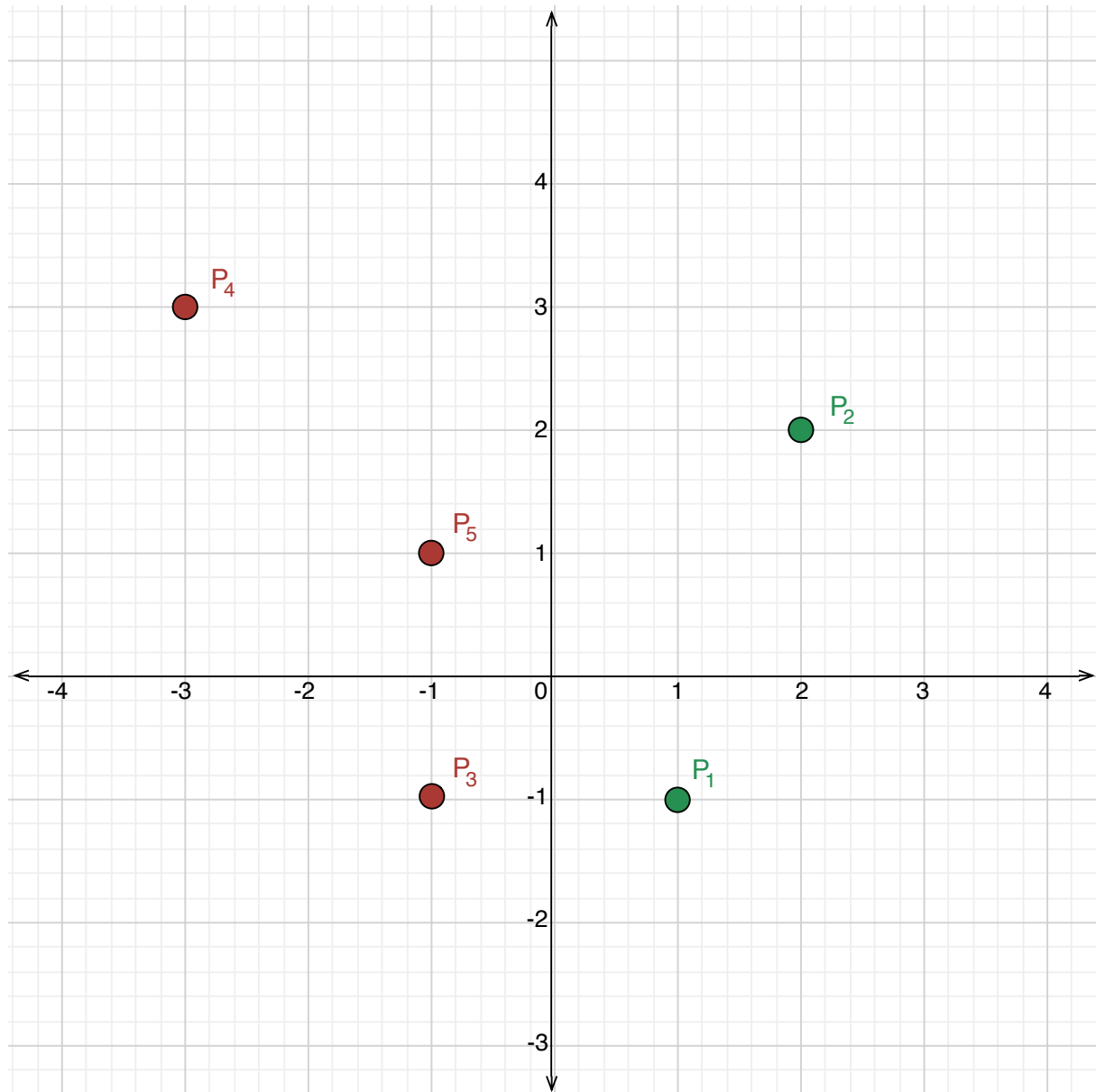


Abbildung 9: Die grünen und roten Punkte repräsentieren die Punkte einer Punktwolke, vor und nach dem Durchgangsfiler.

### Voxel Grid Filter

Die Aufgabe des Voxel Grid Filter (VG-Filter), zu Deutsch Voxeligitter Filter, ist das Entfernen von Punkten einer Punktwolke, wie in Abschnitt 2.6.1 beschrieben. Die Unterteilung der Eingabe-Punktwolke in ein gleichmäßiges 3D-Würfel-Raster ist der erste

Arbeitsschritt des VG-Filter. Die Auflösung des Rasters, sowie die Größe der Würfel sind parametrierbar. Dieses Würfelraster wird als Voxel Grid (VG) bezeichnet [21]. Für die weiteren Arbeitsschritte wird jeder Würfel für sich betrachtet.

Abhängig von der Eingabe-Punktwolke befindet sich innerhalb eines Würfels, eine zufällige Anzahl an Punkten. Um die gefilterte Ausgangs-Punktwolke zu erzeugen wird für jeden Würfel die Koordinate des Schwerpunkts  $P_S$  berechnet. In die Schwerpunktberechnung wird die Anzahl der Punkte  $N$  und deren Position  $x_i$ ,  $y_i$  und  $z_i$  innerhalb des Würfels miteinbezogen.

$$P_S = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad (22)$$

$$P_s = \frac{1}{N} \sum_{i=1}^N P_i = \frac{1}{N} \sum_{i=1}^N \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} \quad (23)$$

Die Punkte der Eingabe-Punktwolke werden entfernt und durch Punkte, welche die berechneten Schwerpunkte aller Würfel repräsentieren, ersetzt [18].

Das folgende theoretische Beispiel veranschaulicht die Funktionsweise eines VG-Filter. Die Ausgangssituation ist im oberen Koordinatensystem der Abbildung 10 veranschaulicht. Die Acht Punkte sind, für das einfachere Verständnis, im 2D-Raum dargestellt. Die Punkte der Eingabe-Punktwolke werden als nicht ausgefüllte Kreise und die berechneten Schwerpunkte als schwarz ausgefüllte Kreise dargestellt.

In Möglichkeit A der Abbildung 10 ist der Schwerpunkt mit Gewichtung aller acht Punkte berechnet. In Möglichkeit B der Abbildung 10 ist ein detaillierteres Raster parametrierbar. Die Schwerpunktberechnung erfolgt für vier Quadrate. Die exemplarische Berechnung des Schwerpunktes  $P_S$  über die Punkte  $P_1$  und  $P_2$ , dargestellt in Abbildung 10 im blau markierten Quadrat, erfolgt in den Schritten  $x_j = \frac{1}{2} \cdot (0+1) = 0,5$  und  $y_j = \frac{1}{2} \cdot (1+2) = 1,5$ . Die berechneten Schwerpunkte  $P_s$  bilden die neue Ausgabe-Punktwolke des VG-Filter. Die Punkte der Eingabe-Punktwolke werden nicht in die Ausgabe-Punktwolke übernommen.

$$P_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, P_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}, P_S = \begin{pmatrix} X_j \\ Y_j \end{pmatrix} \quad (24)$$

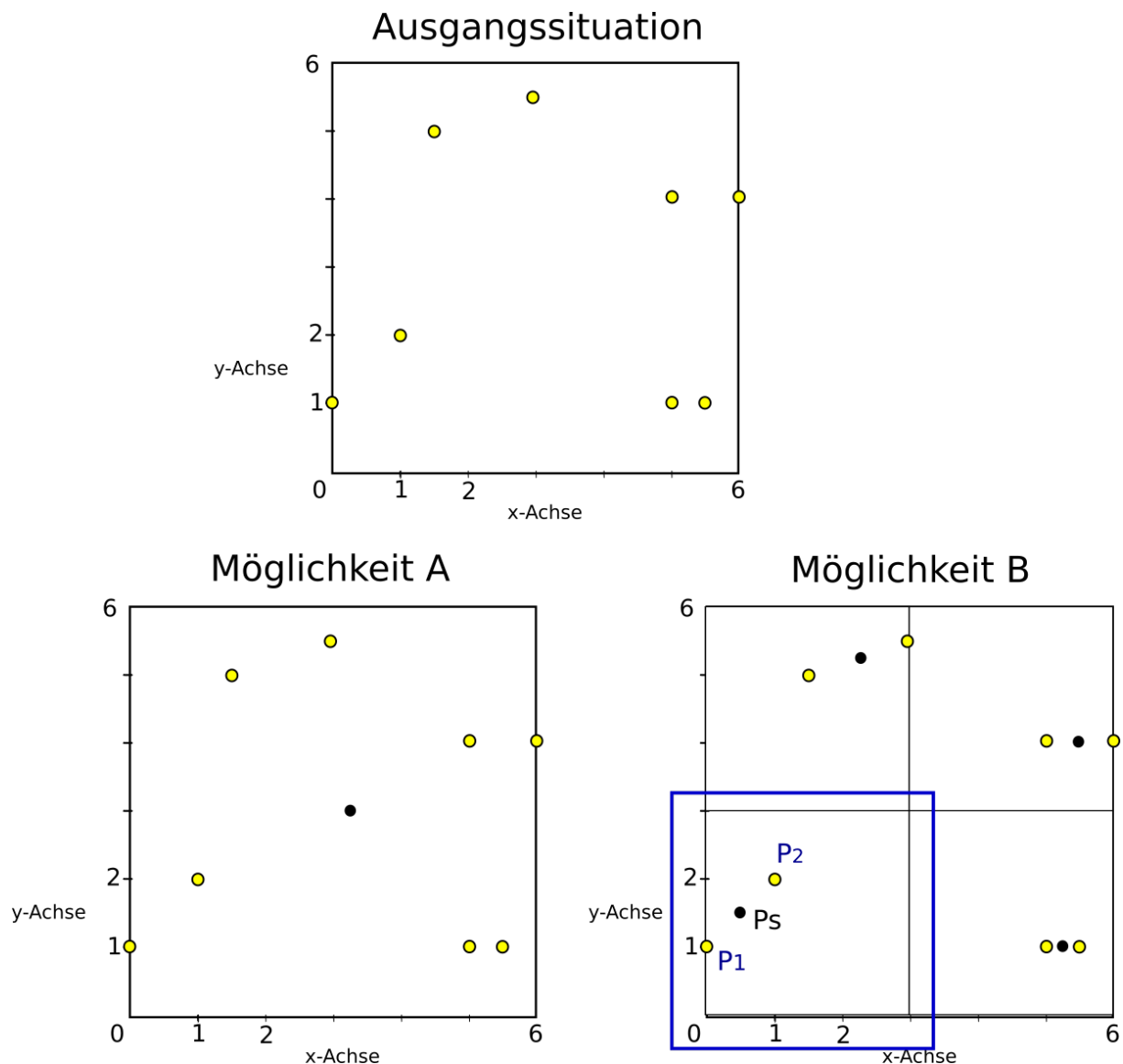


Abbildung 10: Schematische Darstellung des VG-Filter basierend auf [18].

### Statistical Outlier Removal Filter

Die Aufgabe des Statistical Outlier Removal Filter (SOR-Filter), zu Deutsch statistischer Ausreißer-Entfernungsfilter, ist das Bestimmen und Entfernen von Punkten einer Punktwolke, welche als Ausreißer detektiert worden sind; siehe Abschnitt 2.6.1 für eine detaillierte Beschreibung. Im folgenden Absatz wird die Arbeitsweise des SOR-Filters erläutert.

Für die Erzeugung einer Ausgabe-Punktwolke muss der SOR-Filter zweimal über die Eingabe-Punktwolke iterieren. Die Detektierung der Ausreißer wird in der ersten Iteration durchgeführt. In der zweiten Iteration werden die detektierten Ausreißer entfernt. Der SOR-Filter trifft die Annahme, dass die mittlere euklidische Distanz eines Punktes zu seinen Nachbarpunkten normal verteilt ist. In der folgenden Aufzählung werden die zwei Iterationen beschrieben [19]. Die erste Iteration ist in vier Arbeitsschritte unterteilt:

1. Die mittlere euklidischen Distanz  $d_i$  eines Punktes  $P_i$  zu seinen  $k$  Nachbarpunkten berechnen. Die Anzahl der betrachteten  $k$  Nachbarpunkte kann für den Filter parametrisiert werden. Diese Berechnung wird für jeden Punkt in der Punktwolke durchgeführt.
2. Die mittlere euklidischen Distanz  $\mu_d$  über alle  $d_i$  berechnen.
3. Die Standardabweichung  $\sigma$  der Distanzen  $d_i$  berechnen.
4. Den Schwellenwert  $t$  mit der Formel 25 berechnen. Der Multiplikator  $\alpha$  kann für den Filter parametrisiert werden.

$$t = \mu_d + \alpha \cdot \sigma \quad (25)$$

Die zweite Iteration erzeugt die Ausgabe-Punktwolke, indem alle Punkte  $P_i$  aus der Punktwolke entfernt werden für die gilt  $d_i \geq t$ .

### Moving Least Squares

Die Aufgabe des Moving Least Squares (MLS), zu Deutsch Bewegung der kleinsten Quadrate, ist die Flächenrekonstruktion von Objekten in der Punktwolke. Das Least Squares (LS) Verfahren, zu Deutsch kleinste Quadrate Verfahren, dient als Basis für das MLS und Weighted Least Squares (WLS) Verfahren, zu Deutsch gewichtete Methode der kleinsten Quadrate. Im folgenden Abschnitt wird das LS erklärt und die Unterschiede zum WLS und MLS erläutert [20].

Das LS Verfahren sucht eine global gültige Funktion, mit der die Punkte in der Punktwolke bestmöglich approximiert werden. Die Funktion ist abhängig von der Summe  $e_{LS}$  aller quadrierten, horizontalen Abstände der Punkte. Das Ziel ist eine globale Approximation zu ermitteln, die eine möglichst geringe Summe  $e_{LS}$  aufweist [20].

Im Gegensatz zum LS Verfahren, welches eine Funktion über die gesamte Punktwolke sucht, wendet das WLS Verfahren eine lokal gewichtete Approximation an. Für WLS wird eine stationäre Stützstelle festgelegt, die das Zentrum der lokal gewichteten Approximation darstellt. Der Radius des Betrachtungsbereiches kann parametrisiert werden. Punkte die innerhalb des Betrachtungsbereiches liegen werden für die Berechnung der Funktion herangezogen. Alle Punkte, die außerhalb des definierten Bereichs liegen, werden für die Berechnung nicht berücksichtigt. Die lokale Approximation ist für die Punkte innerhalb des parametrisierten Bereiches genauer als die globale Approximation von LS [20].

Im MLS Verfahren ist die im WLS erwähnte Stützstelle und der dazugehörige Betrachtungsbereich nicht stationär, sondern bewegt sich über die gesamte Punktwolke. Das Resultat dieses Verfahrens ist eine globale Approximation mit der Genauigkeit einer lokalen Approximation [20].

## 3 Praktischer Teil

Dieses Kapitel beschreibt zunächst die verwendete Hardware und die gewählte Methodik zur Videoaufnahme des Mensch-Roboter-Arbeitsplatzes. Des Weiteren wird die Methodik zum Angleichen der beiden Punktwolken und die konkrete Implementierung inklusive verwendeter Software beschrieben. Nach der Auswertung der Ergebnisse werden diese zusammengefasst und diskutiert.

### 3.1 Kamerahardware und Versuchsaufbau

Abbildung 12 zeigt den zuvor beschriebenen Mensch-Roboter-Arbeitsplatz. Zur Aufzeichnung dessen werden zwei Intel D435i 3D-Kameras [22] verwendet. Um für den Roboter und den Menschen kein Hindernis darzustellen werden die Kameras auf je einem Kugelkopf und einer Kameraschiene montiert (siehe Abbildung 11). Hierdurch lassen sich beide Kameras nachträglich verschieben und rotieren.

Bei einer Auflösung von 1280x720 Pixel und einer Bildwiederholrate von 30 Hz ergibt sich eine Datenrate von bis zu 885 Mbit/s pro Kamera [23]. Beide Kameras werden daher an den PC direkt über zwei aktive USB 3.1 Kabel mit je 15m Länge angeschlossen.



Abbildung 11: Verwendete Kamera- und Montagehardware

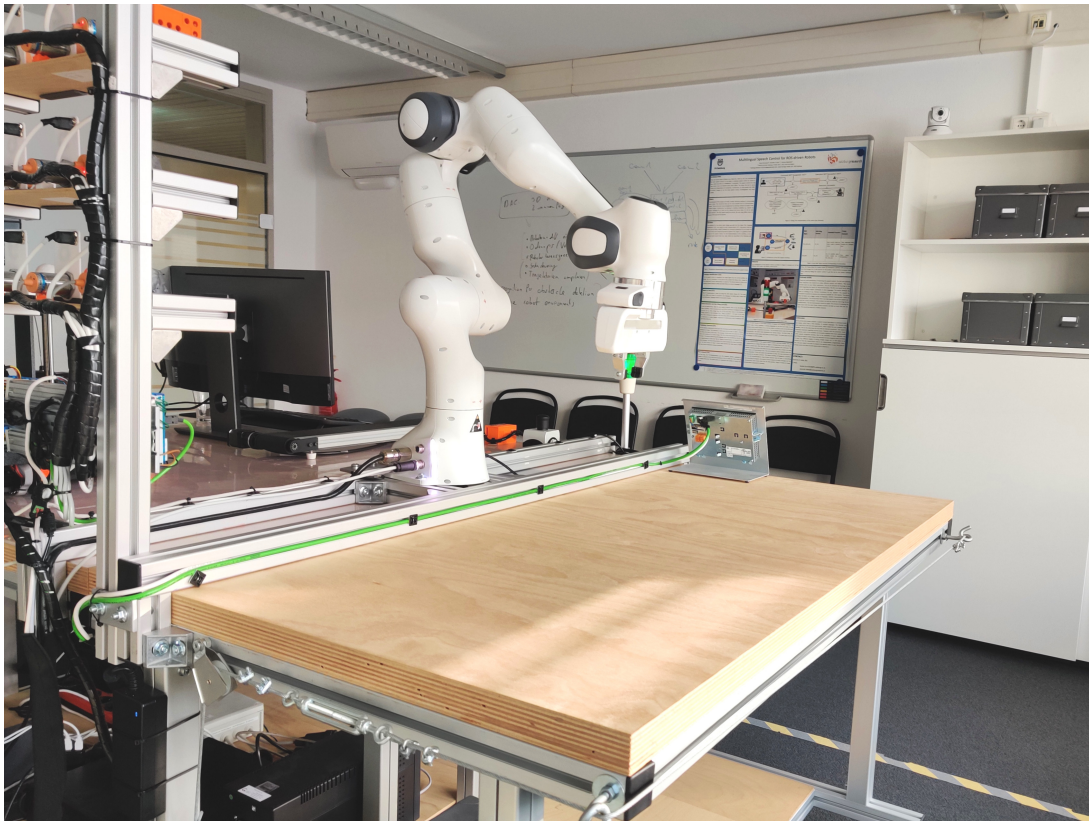


Abbildung 12: Mensch-Roboter-Arbeitsplatz

### 3.1.1 PC-Plattform

Folgende PC-Plattform wird in dieser Arbeit verwendet:

- Central Processing Unit (CPU): Intel I7 8700K
- Random-Access Memory (RAM): 32Gb DDR 4
- Graphics Processing Unit (GPU): Nvidia 2070 Super
- Operating System (OS): Ubuntu 16.04



## 3.2 Implementierung

In diesem Abschnitt werden zuerst die zur Implementierung verwendeten Werkzeuge vorgestellt und anschließend die Testapplikation und der Prototyp beschrieben. Zuletzt werden die verwendeten Datensätze beschrieben und die damit durchgeführten Tests geschildert.

### 3.2.1 Verwendete Werkzeuge

Das Robot Operating System (ROS) ist eine weit verbreitete Basis für die Erstellung von Robotik-Applikationen. ROS ist eigentlich kein Betriebssystem, sondern ein Framework, welches eine einheitliche Kommunikationsschicht für Publisher-Subscriber-Kommunikation zur Verfügung stellt. Daten können dabei auf sogenannte ROS-Topics publiziert und von anderen Applikationen über den Namen des ROS-Topics konsumiert werden. Zusätzlich bietet es eine Menge an fertigen Tools und Bibliotheken, unter anderem für die Übertragung und Visualisierung von Punktwolken [24]. Die für dieses Projekt verwendete Distribution ist *ROS Kinetic Kame*.

Die Point Cloud Library (PCL) ist eine Programmierbibliothek für die Bearbeitung von 3D-Daten. Sie beinhaltet Algorithmen zur Filterung, Registrierung und Werkzeuge zur Visualisierung von Punktwolken. Die PCL ist als C++ Template-Bibliothek realisiert und voll in ROS integriert [25]. Für dieses Projekt wird die Version 1.8.1 verwendet.

### 3.2.2 Testapplikation

Für die im Abschnitt 1.2 definierten Ziele zur Bestimmung der geeigneten Vorverarbeitung und der Auswahl des Algorithmus zum Zusammenführen der Punktwolken wird eine Testapplikation erstellt. Die Testapplikation, dargestellt in Abbildung 13, nimmt zwei Punktwolken als Testdaten entgegen. Als Ausgabe liefert sie den Verlauf des sogenannten Fitness-Scores über die Iterationen des ICP. Der Fitness-Score ist der verbleibende durchschnittliche Fehler nach der Registrierung der Punktwolken, welcher zur Bewertung der Ergebnisse herangezogen wird (siehe Abschnitt 2.4.2). Zusätzlich werden die beiden Punktwolken mit der erstellten Transformation überlagert und dargestellt. Damit kann optisch bewertet werden, ob die Registrierung ein verwertbares Ergebnis erzielt hat.

Wie in Abbildung 13 ersichtlich, führt die Applikation zuerst eine grobe Vorausrichtung durch. Damit soll verhindert werden, dass der ICP an einer falschen Stelle ein lokales Minimum des Fehlers findet [4]. Abbildung 14 stellt beispielhaft zwei Punktwolken in ihren Positionen nach der groben Vorausrichtung dar. Als nächstes werden die im Abschnitt 2.6 beschriebenen Filter angewendet, wobei diese einzeln aktiv oder inaktiv geschaltet werden

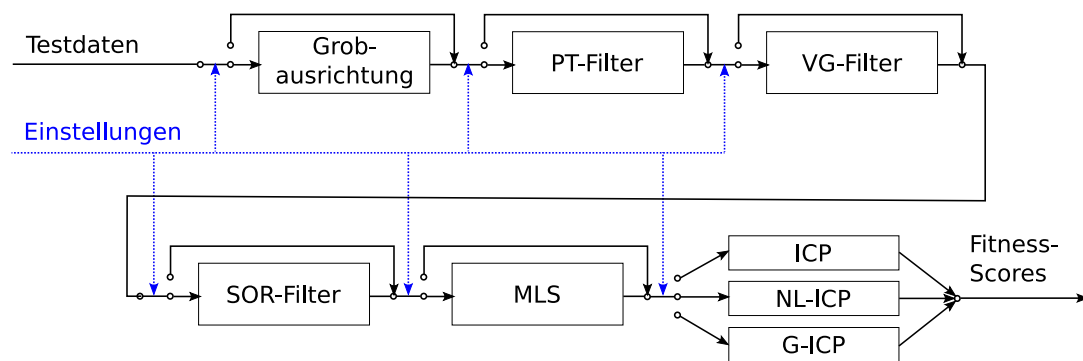


Abbildung 13: Schematische Darstellung des Ablaufs der Testapplikation

können. Letztlich wird eine Registrierung mit einer der in den Abschnitten 2.4 und 2.5 beschriebenen ICP-Varianten durchgeführt. Ein exemplarisches Ergebnis ist in Abbildung 15 dargestellt.

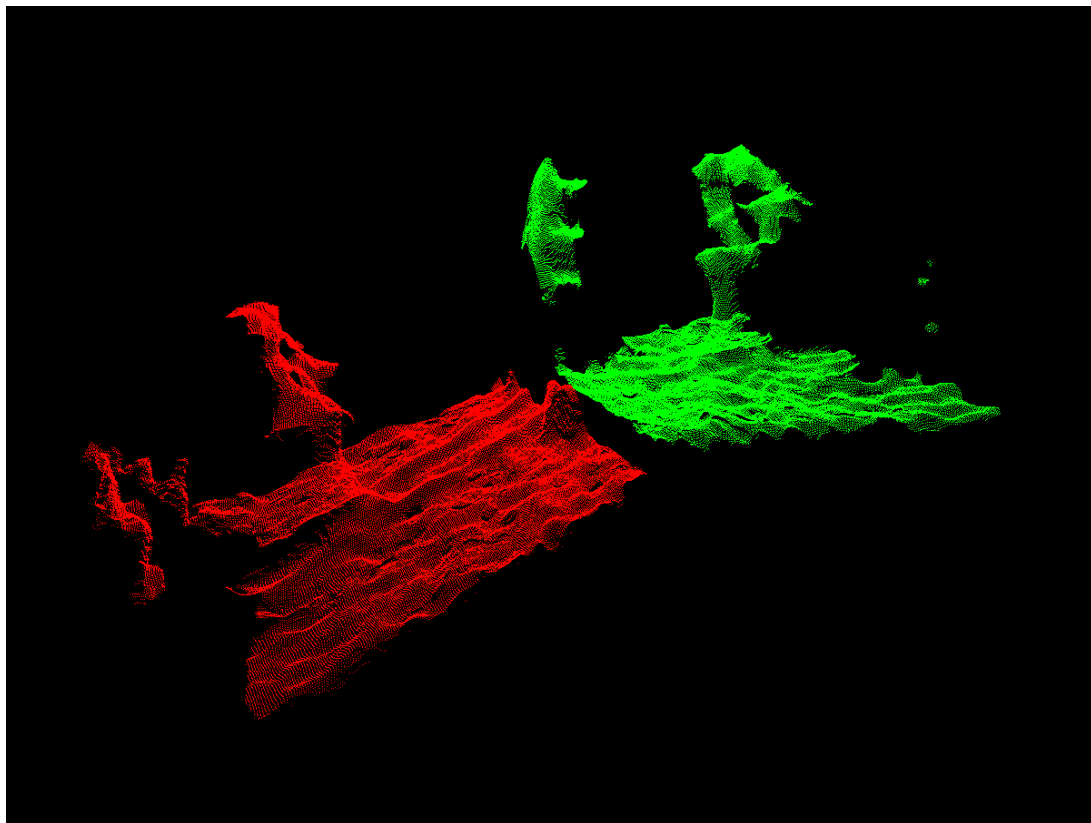


Abbildung 14: Punktwolken nach der groben Vorausrichtung

### 3.2.3 Prototyp

Der Prototyp verwendet für die Vorverarbeitung und Registrierung die Einstellungen, welche in den Tests im Abschnitt 3.4 die besten Ergebnisse erzielt haben. Seine Aufgabe ist

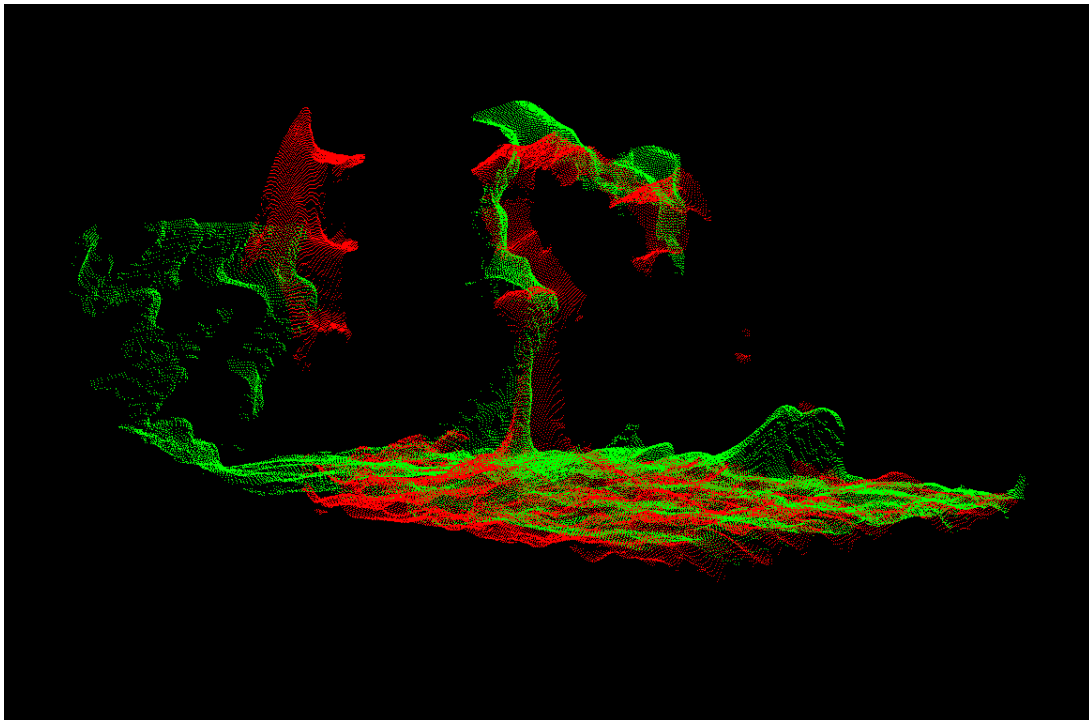


Abbildung 15: Diese Abbildung zeigt ein Ergebnis einer erfolgreichen Registrierung durch den NL-ICP. Die von beiden Kameras aufgenommenen Objekte (Roboterarm und Tisch) werden überlagert. Die Flächen auf der linken Seite sind Teile eines Regals, von dem die zwei Kameras jeweils unterschiedliche Teile aufgenommen haben.

es, einzelne Bilder von 3D-Videodaten aus ROS-Topics zu speichern, damit die Registrierung durchzuführen und die resultierende Transformationsmatrix auf das ROS-Topic `/tf` zu publizieren. Diese Transformationsmatrix kann anschließend verwendet werden, ohne für jedes Bild neu berechnet werden zu müssen. Die Videodaten zur Live-Anzeige können damit auch bei einer Bildwiederholrate von 30 Hz erfolgreich überlagert werden.

### 3.3 Verwendete 3D-Datensätze

Die verwendeten Datensätze sind Aufzeichnungen aus dem im Abschnitt 3.1 vorgestellten Versuchsaufbau. Ein Datensatz besteht aus jeweils einem Bild pro Kamera, welche zum gleichen Zeitpunkt aufgenommen wurden. Zur Abbildung der unterschiedlichen Zustände des Arbeitsplatzes, die zum Zeitpunkt der Registrierung erwartet werden können, werden Datensätze für die folgenden drei Testszenarios erzeugt:

- *Leer*: Im Kamerabereich befindet sich nur die leere Tischplatte mit dem Roboterarm.
- *Überladen*: Auf dem Tisch befinden sich zusätzlich zu dem Roboterarm noch zwei Schachteln (30cm breit, 30cm lang, 25cm hoch und 30cm breit, 40cm lang, 10cm hoch) sowie ein Arbeitshelm.

- *Anlernen*: Im Arbeitsbereich befindet sich eine Person, die dem Roboterarm gerade mit Handführung eine Bewegung beibringt.

## 3.4 Tests

Zur Findung einer geeigneten Programm-Pipeline, werden mit den im Abschnitt 3.3 beschriebenen Datensätzen Tests durchgeführt. Des Weiteren werden die Auswirkungen der Vorverarbeitungsschritte untersucht, indem sie selektiv deaktiviert beziehungsweise optimierte Einstellungen für sie gesucht werden.

### 3.4.1 Auswahl der ICP-Variante

Als erstes werden unterschiedliche ICP-Varianten miteinander verglichen, wobei alle Vorverarbeitungsschritte aktiv sind. Unter den von Bellekens et al. verglichenen ICP-Varianten liefert durchschnittlich der SVD-ICP die besten Ergebnisse [4]. Eine Anwendung des SVD-ICP setzt allerdings voraus, dass schon vor der Registrierung einige Punktkorrespondenzen bekannt sind, was ihn für den Anwendungsfall dieser Arbeit untauglich macht. Es werden daher der Standard-ICP, NL-ICP und G-ICP verglichen.

Abbildung 16 stellt das arithmetische Mittel der Fitness-Scores aller drei Datensätze für die zu vergleichenden ICP-Varianten dar. Der NL-ICP liefert mit 0.0073 den niedrigsten gemittelten Fehler nach 40 Iterationen. Beim Standard-ICP hingegen schlägt die Registrierung beim *Anlernen*-Datensatz fehl, was zu einem entsprechend schlechten Mittelwert führt. Die Testergebnisse sind im Anhang unter A.1 gelistet.

Die Ausführungszeit des Programms unter Verwendung des Standard-ICP ist mit etwa 12 Sekunden merklich schneller als unter Verwendung des NL-ICP oder G-ICP, welche etwa 1 Minute beziehungsweise 1 Minute 40 Sekunden benötigen. Die angegebenen Zeiten sind gemittelte Werte von jeweils drei Messungen, die auf ganze Sekunden gerundet sind.

### 3.4.2 Auswirkungen durch Deaktivierung einzelner Vorverarbeitungsschritte

In diesem Abschnitt werden die prinzipiellen Auswirkungen der einzelnen Vorverarbeitungsschritte bewertet. Dabei wird jeder dieser Schritte einzeln deaktiviert, während alle anderen Schritte aktiv sind. Als ICP-Variante wird der NL-ICP verwendet.

Die Tabelle 1 zeigt die zusammengefassten Ergebnisse, während die Ergebnisse pro Iteration im Anhang unter A.2 gelistet sind. Daraus ist ersichtlich, dass zumindest die Schritte für die grobe Vorausrichtung, PT-Filter und VG-Filter notwendig sind, um zuverlässig zu einem verwertbaren Ergebnis zu kommen. Es ist auch ersichtlich, dass mit dem *Anlernen*-Datensatz auch eine Registrierung ohne PT-Filter beziehungsweise SOR-Filter möglich

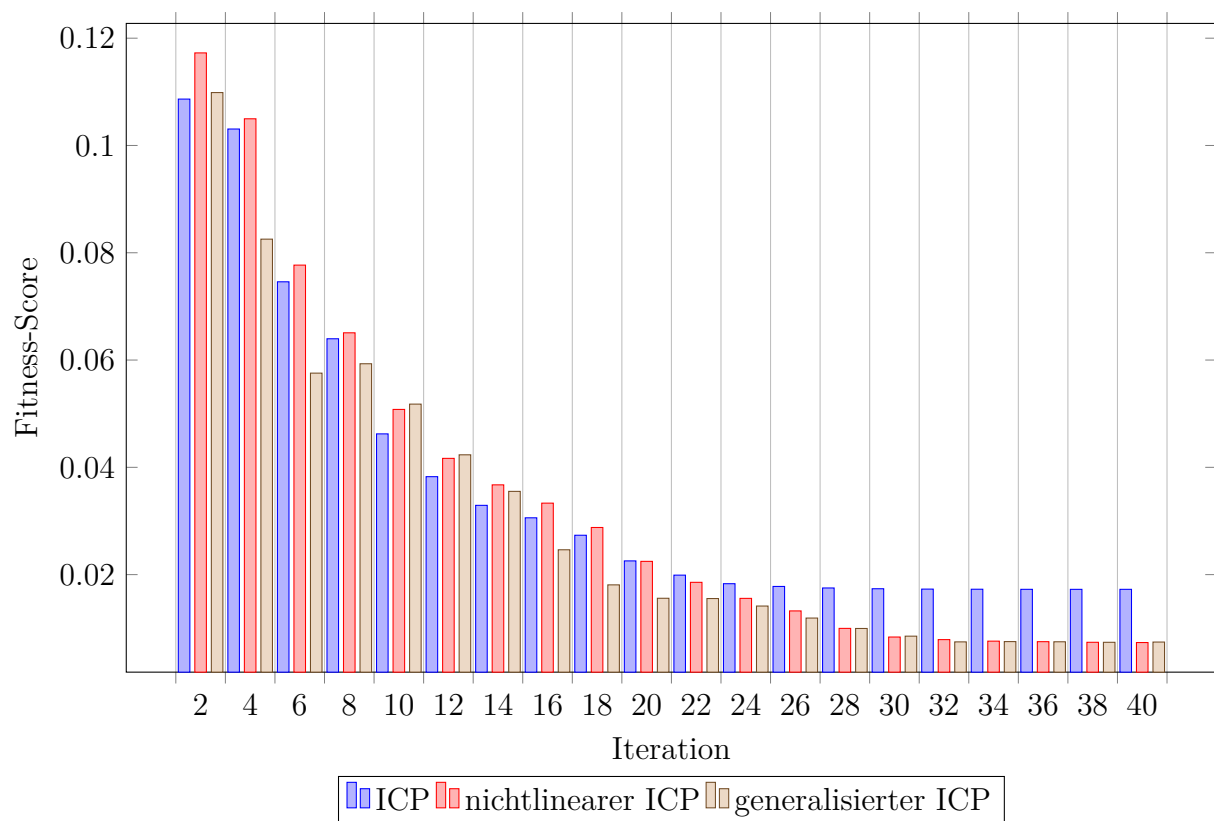


Abbildung 16: Gemittelter Fitness-Score der ICP Varianten im Vergleich

Deaktivierter Schritt	Datensatz	Optische Bewertung	Fitness-Score
grobe Vorausrichtung	Leer	fehlgeschlagen	0.0146
	Überladen	fehlgeschlagen	0.0175
	Anlernen	fehlgeschlagen	0.0317
PT-Filter	Leer	fehlgeschlagen	0.2937
	Überladen	fehlgeschlagen	0.2398
	Anlernen	OK	0.2449
VG-Filter	Leer	fehlgeschlagen	0.0315
	Überladen	fehlgeschlagen	0.0163
	Anlernen	OK	0.0063
SOR-Filter	Leer	OK	0.0064
	Überladen	OK	0.0074
	Anlernen	OK	0.0073
MLS	Leer	OK	0.0053
	Überladen	OK	0.0072
	Anlernen	OK	0.0063

Tabelle 1: Registrierergebnisse nach Deaktivierung einzelner Vorverarbeitungsschritte

ist. Der Grund wird darin vermutet, dass die zusätzliche Person im Bild zu einem höheren Teil an Bildpunkten führt, welche von beiden Kameras erfasst werden. Die Schritte SOR-Filter und MLS hingegen haben nur eine Auswirkung auf die Genauigkeit.

### 3.4.3 Parameterveränderung der einzelnen Vorverarbeitungsschritte

In diesem Abschnitt werden die Auswirkungen der einstellbaren Parameter von den jeweiligen Vorverarbeitungsschritten, welche in Abschnitt 2.6.2 beschrieben wurden, festgehalten. Die Durchführung dieser Bewertung unterliegt folgenden Rahmenbedingungen.

- Der NL-ICP wird, auf Grund der besten erzielten Ergebnisse, für die Punktwolken-Registrierung verwendet.
- Für jeden Test wird exakt ein Parameter verändert. Nach Abschluss des Testes wird der Standardwert wiederhergestellt.
- Parameter, die nicht explizit angegeben werden sind nicht verändert worden und besitzen den Wert der Werkseinstellung.
- Die angegebenen Iterationen beziehen sich auf den NL-ICP. Die Vorverarbeitungsschritte werden einmal ausgeführt.

Die Bedeutung und Auswirkung der Parameter werden in den folgenden Abschnitten des jeweiligen Vorverarbeitungsschrittes erklärt.

#### Parameteroptimierung des VG Filters

Für den VG Filter wird das Format eines Würfels im Raster parametrisiert, beschrieben in Abschnitt 2.6.2. Im Quellcode ist der Parameter als „LeafSize (0.007, 0.007, 0.007)“ definiert. Die Werkseinstellungen für alle drei Übergabeparameter betragen 0.03. Sie bestimmen Länge, Breite und Höhe der Würfel und werden in der Einheit cm angegeben. Die Kameras liefern mit Werkseinstellungen mm-Werte als Punktwolkenkoordinaten, siehe [26].

Der Fitness-Score der Punktwolken-Registrierung mit dem optimierten Parameter des VG Filters ist im Vergleich zu einer Punktwolken-Registrierung mit Standardwerten des VG Filters bei zwei Iterationen besser, siehe Abbildung 17. Die Punktwolken-Registrierung mit Standardwerten erzielt in den darauffolgenden Iterationen ein besseres Ergebnis. Die Punktwolken-Registrierung mit optimiertem Parameter erzielt ab Iteration 42 einen niedrigeren Fitness-Score wie mit dem Standardwert. Mit der Optimierung des Parameters werden die in Abschnitt 2.6.1 beschriebenen Auswirkungen des VG Filters verbessert.

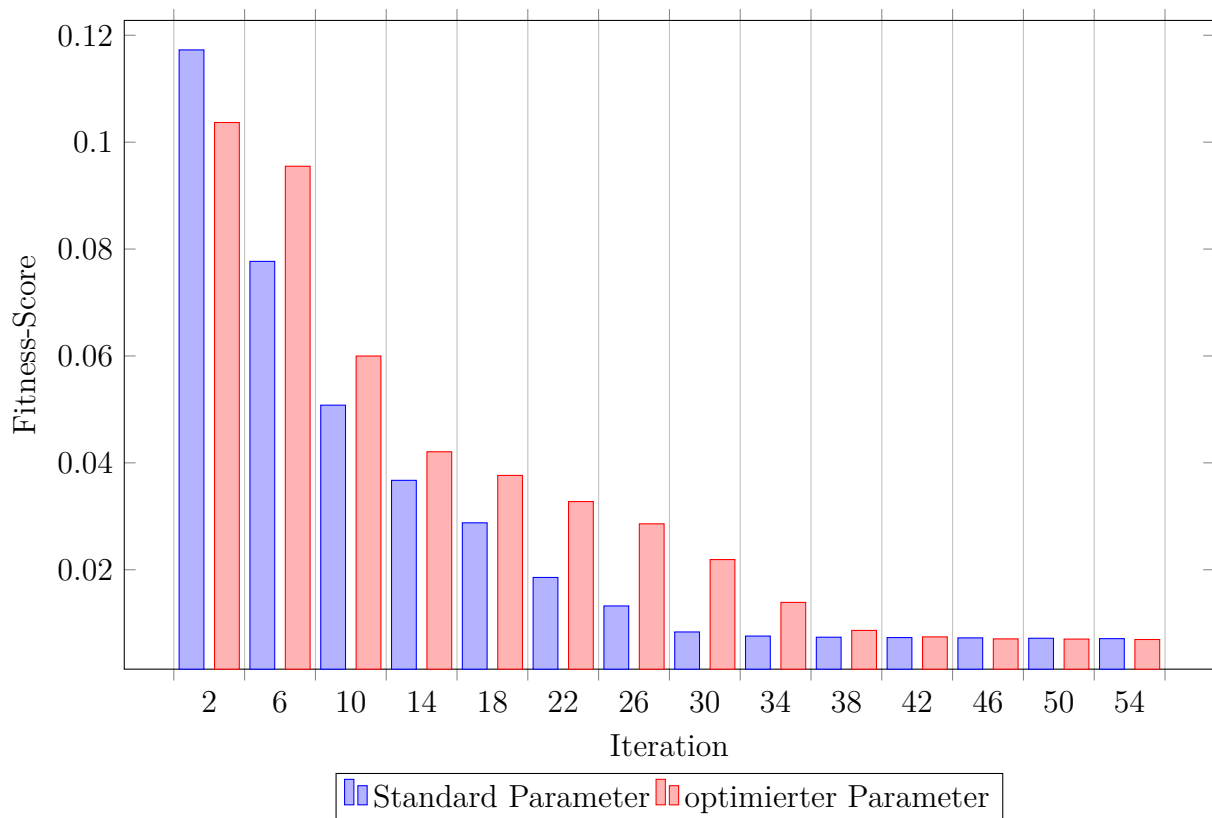


Abbildung 17: Auswirkungen des optimierten Parameters für VG Filter auf Fitness-Score und Iterationen

### Parameteroptimierung des SOR Filters

Für den SOR Filter wird die Anzahl an betrachteten Nachbarpunkten auf 200 parametrisiert. Die Auswirkungen dieses Parameters werden in Abschnitt 2.6.2 erklärt. Die Auflösung mit der die Punktwolken übertragen werden beträgt 640 x 480 und ergibt 307200 Punkte. Die Auflösung kann über Programmparameter justiert werden [26]. Im Quellcode ist der Parameter als „MeanK = 200“ definiert. Die Werkseinstellung für diesen Übergabeparameter beträgt 50. Für diesen Parameter wird keine Einheit benötigt, der Wert gibt die Anzahl der Nachbarpunkte an.

Der Fitness-Score der Punktwolken-Registrierung mit dem optimierten Parameter des SOR Filters ist im Vergleich zur Punktwolken-Registrierung mit Standardwerten des SOR Filters bis zu Iteration 30 schlechter (höherer Wert wird erreicht). Ab Iteration 34 wird mit dem optimierten Parameter ein besserer Fitness-Score erreicht als mit dem Standard Parameter, siehe Abbildung 18. Mit der Optimierung des Parameters werden die in Abschnitt 2.6.1 beschriebenen Auswirkungen des „StatisticalOutlierRemoval filter“ verbessert.

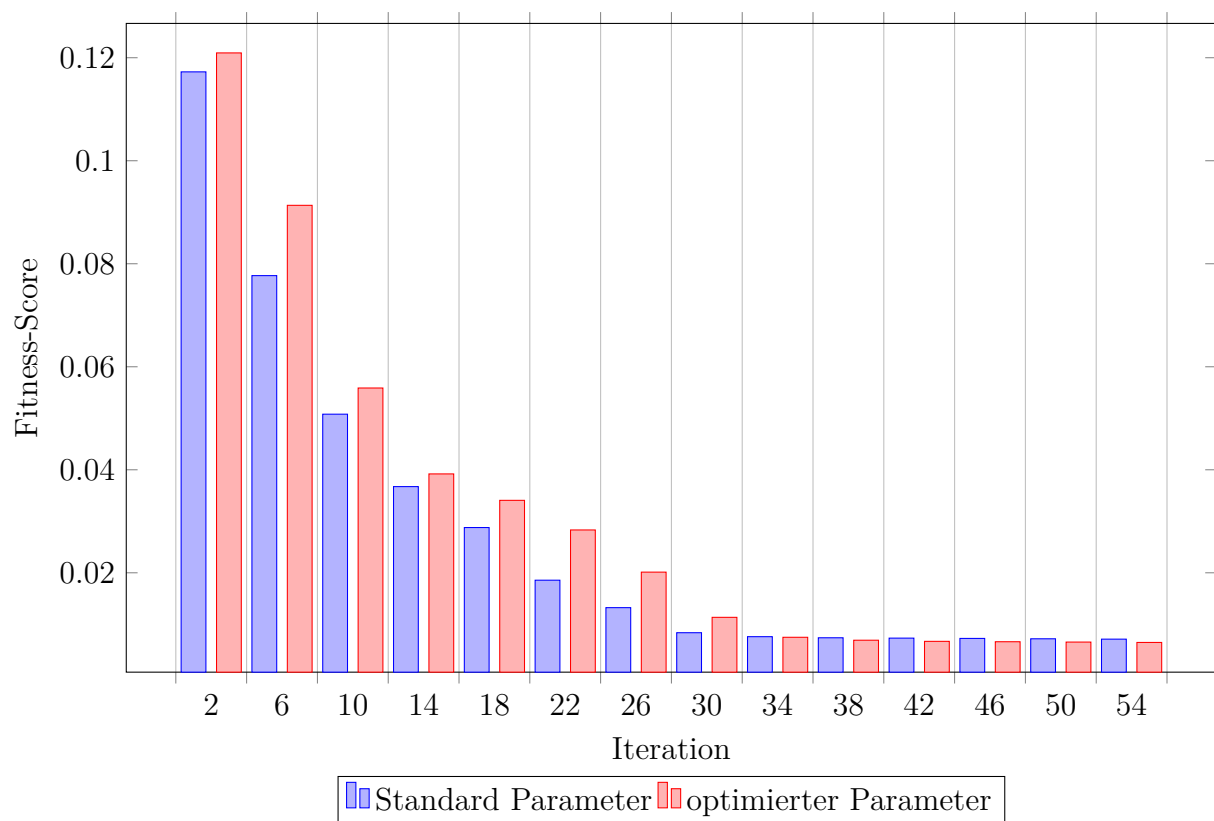


Abbildung 18: Auswirkungen des optimierten Parameters für SOR Filter auf Fitness-Score und Iterationen

### Parameteroptimierung des MLS

Für den MLS Vorverarbeitungsschritt wird der Radius des betrachteten Bereichs einer Approximation auf 0.20 parametrisiert. Die Werkseinstellung für diesen Übergabeparameter beträgt 0.03 und besitzt die Einheit cm. Die Auswirkungen dieses Parameters werden in Abschnitt 2.6.2 erklärt. Im Quellcode ist der Parameter als „setSearchRadius (0.20)“ definiert.

Der Fitness-Score der Punktwolken-Registrierung mit dem optimierten Parameter des MLS ist, mit Ausnahme von Iterationen 6 und 8, höher als bei der Punktwolken-Registrierung mit Standardwerten. Ab Iteration 50 nähert sich der Fitness-Score dem optimierten Parameter, dem Fitness-Score der Punktwolken-Registrierung mit Standardwerten an, erreicht den Wert allerdings nicht, siehe Abbildung 19.

Mit der Optimierung des Parameters werden die in Abschnitt 2.6.1 beschriebenen Auswirkungen des MLS Vorverarbeitungsschrittes optimiert. Die Auswirkungen des MLS auf die Ausgabe-Punktwolke, siehe Abschnitt 2.6.1, machen eine Verschlechterung des Fitness-Scores vernachlässigbar.



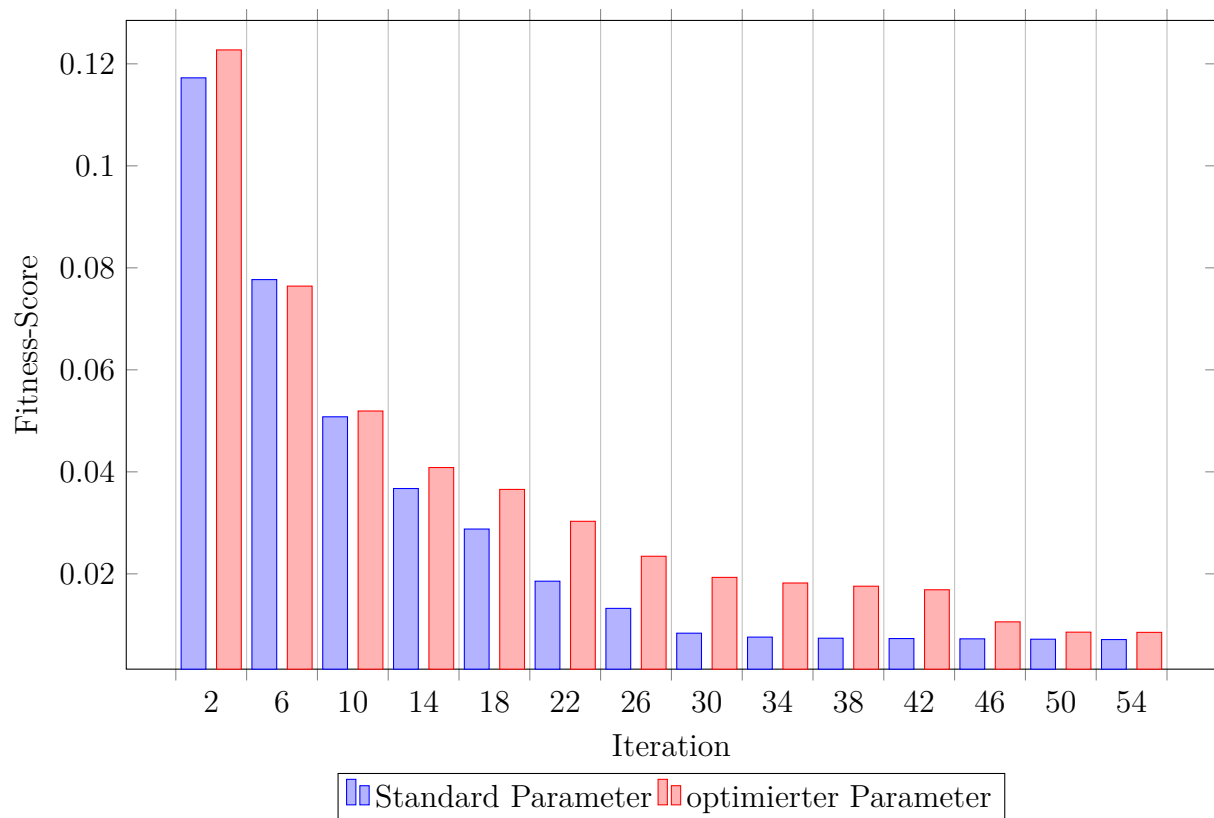


Abbildung 19: Auswirkungen des optimierten Parameters für MLS auf Fitness-Score und Iterationen

### Parameteroptimierung aller Vorverarbeitungsschritte

In diesem Testlauf ist die Parameteroptimierung an allen Vorverarbeitungsschritten gemeinsam durchgeführt worden.

Der Fitness-Score der Punktwolken-Registrierung mit den optimierten Parametern ist ab Iteration 10 höher als der Fitness-Score mit Standardwerten, siehe Abbildung 20. Der Fitness-Score der optimierten Vorverarbeitungen nähert sich dem Ergebnis mit den Standardwerten an, erreicht den Fitness-Score allerdings nicht. Durch die Verwendung des MLS und der daraus resultierenden Erhöhung des Fitness-Scores lässt sich dieses Ergebnis nachvollziehen.

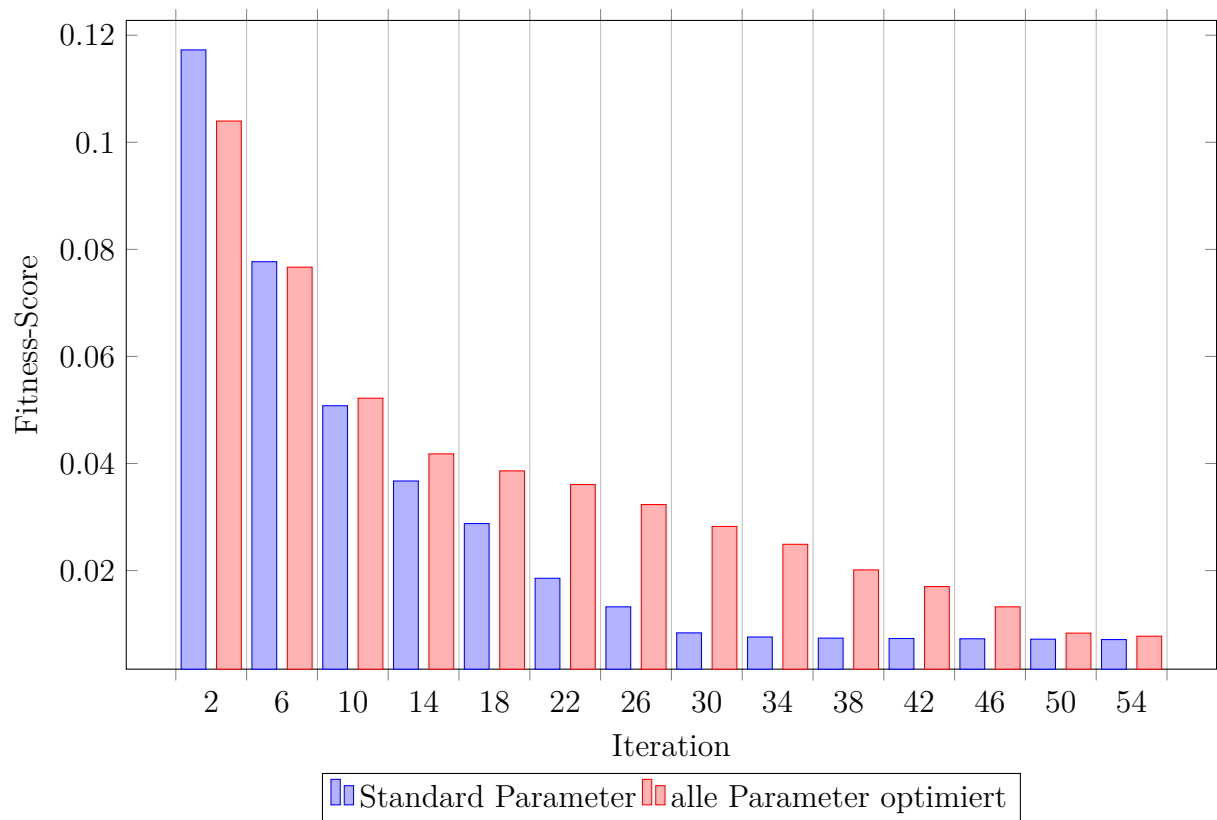


Abbildung 20: Auswirkungen der optimierten Parameter aller Vorverarbeitungsschritte auf Fitness-Score und Iterationen

## 4 Fazit und Ausblick

Die durchgeführten Tests zeigen, dass sowohl der NL-ICP als auch der G-ICP für die Problemstellung geeignet sind, wobei der NL-ICP zu einem leicht besseren Ergebnis kommt. Der Standard-ICP ist ungeeignet, da er nicht für alle Eingangsdaten zu einem verwertbaren Ergebnis kommt. Seine zeitlich bessere Performance ist nicht relevant, da die Transformation nur nach manuellem Umstellen der Kameras neu berechnet werden muss. Durch die Lösung der Transformationen über das ROS-Topic `/tf` ist ein Zusammenführen bei einer Bildwiederholrate von 30 Hz möglich.

Von den in dieser Arbeit getesteten Algorithmen zeigte in [4] der Standard-ICP die besten und der G-ICP die schlechtesten Ergebnisse für die Robustheit. Mit den hier verwendeten Testdaten, zeigt der NL-ICP die besten Ergebnisse, während der Standard-ICP die wenigsten verwertbaren Ergebnisse liefert.

Bei der Vorverarbeitung zeigt sich, dass zumindest eine grobe Vorausrichtung, Voxel Grid Filter und ein Pass Through Filter notwendig sind, um zuverlässig verwertbare Ergebnisse zu bekommen. Ebenfalls wird mit der Parameteroptimierung der Vorverarbeitungsschritte ein verbessertes Ergebnis der Ausgabe-Punktwolke erzielt.

Die Implementierung dieser Arbeit kann als Grundlage für Arbeiten in Richtung der Optimierung der Bewegungen des Roboters dienen. Das verwendete Preprocessing wurde in dieser Arbeit nur auf zwei extrahierte Bilder angewandt um eine Transformationsmatrix anschließend auf die Live-Aufnahmen der Kameras anzuwenden. Für eine zukünftige Kollisionserkennung kann lediglich ein Live-Filtern der Punktwolken Sinn ergeben. Hierfür eignet sich die Implementierung jedoch aufgrund der Geschwindigkeit des Filter-Prozesses nicht. Eine Möglichkeit zur Lösung dieses Problems kann die Berechnung über die hierfür schnellere GPU anstatt der CPU sein.

# Anhang

## A Messergebnisse

Nachfolgend sind die Tabellen mit den Messergebnissen der im Abschnitt 3.4 durchgeführten Tests. Sie zeigen immer für jede zweite Iteration des ICP den Fitness-Score für alle drei der im Abschnitt 3.3 beschriebenen Datensätze sowie deren Mittelwert.

### A.1 ICP-Varianten unter Verwendung aller Vorverarbeitungsschritte

Nachfolgend sind die Tabellen mit den Messergebnissen der im Abschnitt 3.4.1 durchgeführten Tests.

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1408	0.1275	0.0576	0.1086
4	0.0967	0.0908	0.1217	0.1031
6	0.0524	0.0600	0.1114	0.0746
8	0.0498	0.0471	0.0950	0.0640
10	0.0444	0.0336	0.0607	0.0462
12	0.0406	0.0243	0.0499	0.0383
14	0.0373	0.0204	0.0411	0.0329
16	0.0329	0.0190	0.0399	0.0306
18	0.0257	0.0170	0.0393	0.0273
20	0.0145	0.0144	0.0388	0.0226
22	0.0094	0.0117	0.0386	0.0199
24	0.0065	0.0100	0.0384	0.0183
26	0.0060	0.0092	0.0382	0.0178
28	0.0060	0.0086	0.0380	0.0175
30	0.0059	0.0083	0.0379	0.0174
32	0.0059	0.0082	0.0378	0.0173
34	0.0059	0.0081	0.0378	0.0173
36	0.0059	0.0081	0.0378	0.0173
38	0.0059	0.0081	0.0378	0.0173
40	0.0059	0.0081	0.0378	0.0173
42	0.0059	0.0081	0.0378	0.0173
44	0.0059	0.0081	0.0378	0.0173
46	0.0059	0.0081	0.0378	0.0173
48	0.0059	0.0081	0.0378	0.0173
50	0.0059	0.0081	0.0378	0.0173
52	0.0059	0.0081	0.0378	0.0173
54	0.0059	0.0081	0.0378	0.0172
56	0.0059	0.0081	0.0378	0.0172
58	0.0059	0.0081	0.0378	0.0172
60	0.0059	0.0081	0.0378	0.0172

Tabelle 2: Ergebnisse des Standard-ICP unter Verwendung aller Vorverarbeitungsschritte

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1105	0.1224	0.1188	0.1173
4	0.0790	0.1046	0.1314	0.1050
6	0.0525	0.0754	0.1052	0.0777
8	0.0454	0.0579	0.0919	0.0651
10	0.0403	0.0411	0.0710	0.0508
12	0.0361	0.0280	0.0609	0.0417
14	0.0318	0.0226	0.0559	0.0367
16	0.0279	0.0198	0.0523	0.0333
18	0.0205	0.0184	0.0474	0.0288
20	0.0095	0.0166	0.0414	0.0225
22	0.0066	0.0146	0.0345	0.0186
24	0.0061	0.0117	0.0289	0.0156
26	0.0060	0.0100	0.0237	0.0132
28	0.0060	0.0092	0.0147	0.0100
30	0.0060	0.0086	0.0105	0.0084
32	0.0060	0.0083	0.0094	0.0079
34	0.0059	0.0082	0.0087	0.0076
36	0.0059	0.0081	0.0084	0.0075
38	0.0059	0.0081	0.0081	0.0074
40	0.0059	0.0081	0.0080	0.0073
42	0.0059	0.0081	0.0080	0.0073
44	0.0059	0.0081	0.0077	0.0072
46	0.0059	0.0081	0.0078	0.0073
48	0.0059	0.0081	0.0076	0.0072
50	0.0059	0.0081	0.0076	0.0072
52	0.0059	0.0081	0.0075	0.0072
54	0.0059	0.0081	0.0074	0.0071
56	0.0059	0.0081	0.0074	0.0071
58	0.0059	0.0081	0.0073	0.0071
60	0.0059	0.0081	0.0073	0.0071

Tabelle 3: Ergebnisse des NL-ICP unter Verwendung aller Vorverarbeitungsschritte

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1023	0.1079	0.1193	0.1099
4	0.0701	0.0967	0.0809	0.0825
6	0.0382	0.0624	0.0721	0.0576
8	0.0340	0.0646	0.0794	0.0593
10	0.0412	0.0466	0.0676	0.0518
12	0.0395	0.0336	0.0539	0.0423
14	0.0298	0.0298	0.0469	0.0355
16	0.0214	0.0333	0.0192	0.0246
18	0.0089	0.0334	0.0120	0.0181
20	0.0063	0.0329	0.0075	0.0156
22	0.0062	0.0330	0.0074	0.0155
24	0.0062	0.0283	0.0079	0.0141
26	0.0061	0.0217	0.0079	0.0119
28	0.0061	0.0160	0.0079	0.0100
30	0.0060	0.0120	0.0076	0.0085
32	0.0060	0.0087	0.0077	0.0075
34	0.0062	0.0087	0.0077	0.0075
36	0.0061	0.0087	0.0077	0.0075
38	0.0061	0.0084	0.0077	0.0074
40	0.0061	0.0085	0.0077	0.0074
42	0.0061	0.0085	0.0077	0.0074
44	0.0061	0.0085	0.0077	0.0074
46	0.0061	0.0085	0.0077	0.0074
48	0.0061	0.0085	0.0077	0.0074
50	0.0061	0.0085	0.0077	0.0074
52	0.0061	0.0084	0.0076	0.0074
54	0.0061	0.0084	0.0077	0.0074
56	0.0061	0.0084	0.0077	0.0074
58	0.0061	0.0084	0.0077	0.0074
60	0.0061	0.0084	0.0077	0.0074

Tabelle 4: Ergebnisse des G-ICP unter Verwendung aller Vorverarbeitungsschritte

## A.2 NL-ICP bei deaktivierten Vorverarbeitungsschritten

Nachfolgend sind die Tabellen mit den Messergebnissen der im Abschnitt 3.4.2 durchgeführten Tests.

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.0533	0.0321	0.0457	0.0437
4	0.0360	0.0292	0.0405	0.0352
6	0.0300	0.0271	0.0386	0.0319
8	0.0228	0.0252	0.0365	0.0282
10	0.0199	0.0238	0.0351	0.0263
12	0.0190	0.0228	0.0339	0.0253
14	0.0186	0.0220	0.0332	0.0246
16	0.0181	0.0211	0.0329	0.0240
18	0.0175	0.0204	0.0325	0.0235
20	0.0171	0.0196	0.0325	0.0231
22	0.0166	0.0192	0.0322	0.0227
24	0.0161	0.0190	0.0320	0.0224
26	0.0159	0.0189	0.0319	0.0222
28	0.0157	0.0187	0.0318	0.0221
30	0.0155	0.0186	0.0316	0.0219
32	0.0154	0.0184	0.0315	0.0218
34	0.0154	0.0184	0.0314	0.0217
36	0.0151	0.0182	0.0315	0.0216
38	0.0151	0.0182	0.0315	0.0216
40	0.0150	0.0181	0.0315	0.0215
42	0.0148	0.0180	0.0315	0.0214
44	0.0149	0.0180	0.0315	0.0214
46	0.0148	0.0179	0.0315	0.0214
48	0.0148	0.0178	0.0316	0.0214
50	0.0147	0.0178	0.0316	0.0214
52	0.0146	0.0177	0.0316	0.0213
54	0.0146	0.0177	0.0316	0.0213
56	0.0146	0.0177	0.0317	0.0213
58	0.0146	0.0176	0.0317	0.0213
60	0.0146	0.0175	0.0317	0.0213

Tabelle 5: Ergebnisse des NL-ICP ohne grobe Vorausrichtung



Iteration	Leer	Überladen	Anlernen	Mittelwert
2	1.1671	0.9046	0.6427	0.9048
4	1.0471	0.8268	0.5651	0.8130
6	0.9706	0.7594	0.5055	0.7452
8	0.9314	0.7167	0.4848	0.7109
10	0.9075	0.6585	0.4696	0.6786
12	0.8569	0.6192	0.4550	0.6437
14	0.8181	0.5724	0.4400	0.6102
16	0.7790	0.5443	0.4167	0.5800
18	0.7271	0.5143	0.3836	0.5416
20	0.6718	0.4780	0.3517	0.5005
22	0.6102	0.4337	0.3038	0.4492
24	0.5560	0.4001	0.2835	0.4132
26	0.5226	0.3746	0.2712	0.3895
28	0.4906	0.3541	0.2626	0.3691
30	0.4598	0.3342	0.2578	0.3506
32	0.4386	0.3228	0.2538	0.3384
34	0.4190	0.3112	0.2513	0.3271
36	0.4008	0.3005	0.2487	0.3167
38	0.3865	0.2915	0.2472	0.3084
40	0.3728	0.2823	0.2456	0.3002
42	0.3608	0.2743	0.2454	0.2935
44	0.3490	0.2695	0.2452	0.2879
46	0.3391	0.2637	0.2451	0.2826
48	0.3315	0.2578	0.2450	0.2781
50	0.3242	0.2536	0.2446	0.2741
52	0.3171	0.2488	0.2449	0.2703
54	0.3106	0.2466	0.2449	0.2674
56	0.3037	0.2441	0.2449	0.2643
58	0.2988	0.2416	0.2449	0.2618
60	0.2937	0.2398	0.2449	0.2595

Tabelle 6: Ergebnisse des NL-ICP ohne PT-Filter

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.0959	0.0677	0.1218	0.0952
4	0.1079	0.0556	0.1489	0.1041
6	0.0961	0.0474	0.1290	0.0908
8	0.0740	0.0387	0.1115	0.0747
10	0.0536	0.0319	0.0976	0.0610
12	0.0415	0.0284	0.0805	0.0501
14	0.0393	0.0252	0.0632	0.0425
16	0.0391	0.0221	0.0582	0.0398
18	0.0370	0.0207	0.0476	0.0351
20	0.0358	0.0196	0.0413	0.0322
22	0.0352	0.0190	0.0378	0.0307
24	0.0346	0.0187	0.0324	0.0286
26	0.0342	0.0185	0.0288	0.0272
28	0.0340	0.0183	0.0234	0.0252
30	0.0337	0.0181	0.0202	0.0240
32	0.0335	0.0180	0.0194	0.0237
34	0.0335	0.0179	0.0193	0.0235
36	0.0333	0.0178	0.0190	0.0234
38	0.0331	0.0177	0.0187	0.0232
40	0.0330	0.0176	0.0174	0.0227
42	0.0328	0.0175	0.0150	0.0218
44	0.0327	0.0174	0.0123	0.0208
46	0.0326	0.0173	0.0107	0.0202
48	0.0324	0.0171	0.0095	0.0197
50	0.0324	0.0170	0.0080	0.0191
52	0.0323	0.0168	0.0068	0.0186
54	0.0319	0.0166	0.0065	0.0183
56	0.0318	0.0165	0.0064	0.0182
58	0.0317	0.0164	0.0064	0.0182
60	0.0315	0.0163	0.0063	0.0180

Tabelle 7: Ergebnisse des NL-ICP ohne VG-Filter

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1184	0.1258	0.1281	0.1241
4	0.1093	0.1074	0.1236	0.1134
6	0.0851	0.0705	0.0905	0.0820
8	0.0634	0.0579	0.0697	0.0637
10	0.0490	0.0397	0.0631	0.0506
12	0.0415	0.0268	0.0595	0.0426
14	0.0393	0.0219	0.0575	0.0395
16	0.0371	0.0187	0.0545	0.0368
18	0.0346	0.0172	0.0514	0.0344
20	0.0323	0.0146	0.0477	0.0315
22	0.0306	0.0116	0.0424	0.0282
24	0.0288	0.0093	0.0389	0.0257
26	0.0267	0.0083	0.0352	0.0234
28	0.0230	0.0079	0.0281	0.0196
30	0.0147	0.0076	0.0165	0.0129
32	0.0093	0.0075	0.0094	0.0087
34	0.0076	0.0074	0.0085	0.0079
36	0.0070	0.0074	0.0081	0.0075
38	0.0069	0.0074	0.0081	0.0074
40	0.0068	0.0074	0.0079	0.0073
42	0.0068	0.0074	0.0077	0.0073
44	0.0067	0.0074	0.0076	0.0072
46	0.0067	0.0074	0.0075	0.0072
48	0.0066	0.0074	0.0075	0.0071
50	0.0066	0.0074	0.0074	0.0071
52	0.0065	0.0074	0.0074	0.0071
54	0.0065	0.0074	0.0073	0.0070
56	0.0065	0.0074	0.0073	0.0070
58	0.0064	0.0074	0.0073	0.0070
60	0.0064	0.0074	0.0073	0.0070

Tabelle 8: Ergebnisse des NL-ICP ohne SOR-Filter

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1236	0.1143	0.1291	0.1223
4	0.0992	0.0976	0.1224	0.1064
6	0.0727	0.0673	0.0965	0.0788
8	0.0525	0.0507	0.0747	0.0593
10	0.0448	0.0359	0.0651	0.0486
12	0.0418	0.0244	0.0570	0.0411
14	0.0405	0.0206	0.0530	0.0380
16	0.0374	0.0186	0.0490	0.0350
18	0.0349	0.0171	0.0446	0.0322
20	0.0330	0.0153	0.0389	0.0291
22	0.0295	0.0129	0.0326	0.0250
24	0.0228	0.0106	0.0269	0.0201
26	0.0114	0.0089	0.0239	0.0148
28	0.0071	0.0082	0.0140	0.0097
30	0.0056	0.0076	0.0096	0.0076
32	0.0054	0.0074	0.0083	0.0070
34	0.0054	0.0073	0.0078	0.0068
36	0.0054	0.0073	0.0074	0.0067
38	0.0053	0.0072	0.0072	0.0066
40	0.0053	0.0072	0.0070	0.0065
42	0.0053	0.0072	0.0070	0.0065
44	0.0053	0.0072	0.0068	0.0064
46	0.0053	0.0072	0.0067	0.0064
48	0.0053	0.0072	0.0066	0.0064
50	0.0053	0.0072	0.0065	0.0063
52	0.0053	0.0072	0.0065	0.0063
54	0.0053	0.0072	0.0064	0.0063
56	0.0053	0.0072	0.0064	0.0063
58	0.0053	0.0072	0.0063	0.0063
60	0.0053	0.0072	0.0063	0.0063

Tabelle 9: Ergebnisse des NL-ICP ohne MLS

### A.3 NL-ICP bei optimierten Vorverarbeitungsschritten

Nachfolgend sind die Tabellen mit den Messergebnissen der im Abschnitt 3.4.3 durchgeführten Tests.

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1073	0.1045	0.0992	0.1037
4	0.1011	0.0915	0.1258	0.1062
6	0.0932	0.0773	0.1159	0.0955
8	0.0697	0.0631	0.0967	0.0765
10	0.0581	0.0442	0.0777	0.0600
12	0.0487	0.0330	0.0663	0.0493
14	0.0406	0.0269	0.0588	0.0421
16	0.0404	0.0232	0.0567	0.0401
18	0.0384	0.0213	0.0533	0.0377
20	0.0366	0.0202	0.0495	0.0354
22	0.0347	0.0196	0.0440	0.0328
24	0.0329	0.0188	0.0397	0.0304
26	0.0311	0.0180	0.0367	0.0286
28	0.0290	0.0172	0.0334	0.0265
30	0.0210	0.0158	0.0289	0.0219
32	0.0123	0.0141	0.0264	0.0176
34	0.0082	0.0120	0.0215	0.0139
36	0.0060	0.0111	0.0156	0.0109
38	0.0053	0.0105	0.0102	0.0087
40	0.0051	0.0098	0.0089	0.0079
42	0.0050	0.0093	0.0080	0.0074
44	0.0050	0.0091	0.0075	0.0072
46	0.0049	0.0090	0.0073	0.0071
48	0.0049	0.0091	0.0071	0.0070
50	0.0049	0.0090	0.0072	0.0070
52	0.0049	0.0090	0.0071	0.0070
54	0.0049	0.0090	0.0070	0.0070
56	0.0049	0.0090	0.0070	0.0070
58	0.0049	0.0090	0.0069	0.0069
60	0.0049	0.0090	0.0069	0.0069

Tabelle 10: Ergebnisse des NL-ICP mit optimierten VG-Filter

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1237	0.1272	0.1119	0.1209
4	0.1133	0.1084	0.1250	0.1156
6	0.0977	0.0748	0.1015	0.0913
8	0.0779	0.0563	0.0864	0.0735
10	0.0601	0.0403	0.0672	0.0559
12	0.0518	0.0266	0.0595	0.0460
14	0.0418	0.0204	0.0554	0.0392
16	0.0412	0.0171	0.0519	0.0367
18	0.0383	0.0153	0.0486	0.0341
20	0.0364	0.0136	0.0453	0.0318
22	0.0326	0.0116	0.0408	0.0283
24	0.0282	0.0094	0.0372	0.0250
26	0.0194	0.0084	0.0326	0.0201
28	0.0108	0.0079	0.0264	0.0150
30	0.0067	0.0075	0.0199	0.0114
32	0.0059	0.0071	0.0115	0.0082
34	0.0057	0.0070	0.0098	0.0075
36	0.0056	0.0069	0.0089	0.0071
38	0.0056	0.0067	0.0084	0.0069
40	0.0056	0.0066	0.0083	0.0068
42	0.0055	0.0065	0.0080	0.0067
44	0.0055	0.0065	0.0080	0.0066
46	0.0055	0.0064	0.0079	0.0066
48	0.0055	0.0064	0.0078	0.0066
50	0.0055	0.0064	0.0078	0.0066
52	0.0055	0.0064	0.0076	0.0065
54	0.0055	0.0063	0.0076	0.0065
56	0.0055	0.0063	0.0076	0.0065
58	0.0055	0.0063	0.0076	0.0064
60	0.0055	0.0063	0.0075	0.0064

Tabelle 11: Ergebnisse des NL-ICP mit optimierten SOR-Filter

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1269	0.1109	0.1304	0.1227
4	0.1013	0.0988	0.1247	0.1083
6	0.0679	0.0668	0.0946	0.0764
8	0.0504	0.0513	0.0805	0.0607
10	0.0486	0.0384	0.0688	0.0519
12	0.0448	0.0275	0.0611	0.0445
14	0.0431	0.0236	0.0559	0.0408
16	0.0422	0.0225	0.0520	0.0389
18	0.0403	0.0214	0.0480	0.0366
20	0.0395	0.0193	0.0437	0.0342
22	0.0385	0.0166	0.0358	0.0303
24	0.0378	0.0132	0.0296	0.0269
26	0.0371	0.0110	0.0222	0.0235
28	0.0368	0.0099	0.0157	0.0208
30	0.0361	0.0094	0.0124	0.0193
32	0.0360	0.0092	0.0107	0.0186
34	0.0355	0.0092	0.0099	0.0182
36	0.0350	0.0092	0.0095	0.0179
38	0.0344	0.0092	0.0092	0.0176
40	0.0339	0.0092	0.0089	0.0173
42	0.0327	0.0092	0.0088	0.0169
44	0.0287	0.0092	0.0088	0.0155
46	0.0140	0.0092	0.0086	0.0106
48	0.0096	0.0092	0.0086	0.0091
50	0.0080	0.0092	0.0086	0.0086
52	0.0079	0.0092	0.0085	0.0085
54	0.0079	0.0092	0.0085	0.0085
56	0.0079	0.0092	0.0085	0.0085
58	0.0079	0.0092	0.0085	0.0086
60	0.0079	0.0092	0.0085	0.0085

Tabelle 12: Ergebnisse des NL-ICP mit optimierten MLS

Iteration	Leer	Überladen	Anlernen	Mittelwert
2	0.1028	0.0933	0.1158	0.1040
4	0.0936	0.0887	0.1130	0.0984
6	0.0736	0.0671	0.0894	0.0767
8	0.0573	0.0511	0.0768	0.0617
10	0.0490	0.0377	0.0699	0.0522
12	0.0418	0.0293	0.0634	0.0448
14	0.0415	0.0235	0.0604	0.0418
16	0.0390	0.0212	0.0595	0.0399
18	0.0380	0.0203	0.0576	0.0386
20	0.0364	0.0197	0.0562	0.0374
22	0.0351	0.0194	0.0537	0.0361
24	0.0339	0.0190	0.0507	0.0345
26	0.0334	0.0185	0.0451	0.0323
28	0.0328	0.0180	0.0405	0.0304
30	0.0322	0.0173	0.0352	0.0282
32	0.0317	0.0167	0.0298	0.0261
34	0.0311	0.0160	0.0277	0.0249
36	0.0308	0.0155	0.0201	0.0221
38	0.0304	0.0146	0.0154	0.0201
40	0.0297	0.0133	0.0123	0.0184
42	0.0284	0.0125	0.0101	0.0170
44	0.0266	0.0120	0.0090	0.0159
46	0.0199	0.0113	0.0085	0.0132
48	0.0097	0.0103	0.0082	0.0094
50	0.0074	0.0095	0.0080	0.0083
52	0.0059	0.0093	0.0080	0.0077
54	0.0059	0.0093	0.0080	0.0077
56	0.0059	0.0093	0.0079	0.0077
58	0.0059	0.0093	0.0079	0.0077
60	0.0059	0.0093	0.0079	0.0077

Tabelle 13: Ergebnisse des NL-ICP mit allen Optimierungen der Vorverarbeitung



## Literaturverzeichnis

- [1] P. J. Besl und N. D. McKay, „Method for registration of 3-D shapes“, in *Sensor Fusion IV: Control Paradigms and Data Structures*, P. S. Schenker, Hrsg., International Society for Optics und Photonics, Bd. 1611, SPIE, 1992, S. 586–606. DOI: 10.1117/12.57955.
- [2] J. Otepka et al., „Georeferenced Point Clouds: A Survey of Features and Point Cloud Management“, Jg. 2, S. 1038–1065, 4 2013, ISSN: 2220-9964. DOI: 10.3390/ijgi2041038.
- [3] M. X. Zhu et al., „Simultaneous Scene Reconstruction and Auto-calibration using Constrained Iterative Closest Point for 3D Depth Sensor Array“, 12th Conference on Computer und Robot Vision, IEEE, 2015, S. 39–45. DOI: doi:10.1109/crv.2015.13.
- [4] B. Bellekens et al., „A Benchmark Survey of Rigid 3D Point Cloud Registration Algorithms“, *International Journal On Advances in Intelligent Systems*, Jg. 8, Nr. 1, 2, S. 118–127, Juni 2015.
- [5] J. A. Bærentzen et al., „3D Surface Registration via Iterative Closest Point (ICP)“, in *Guide to Computational Geometry Processing*, Springer London, 2012, S. 263–275. DOI: 10.1007/978-1-4471-4075-7\_15.
- [6] A. Nuchter, „Semantische dreidimensionale Karten für autonome mobile Roboter“, Diss., Universität Bonn, 2006.
- [7] G. P. Penney et al., „A Stochastic Iterative Closest Point Algorithm (stochastICP)“, Bd. 2208, Okt. 2001, S. 762–769. DOI: 10.1007/3-540-45468-3\_91.
- [8] K. S. Arun et al., „Least-Squares Fitting of Two 3-D Point Sets“, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Jg. 9, Nr. 5, S. 698–700, Sep. 1987, ISSN: 1939-3539. DOI: 10.1109/TPAMI.1987.4767965.
- [9] O. Sorkine-Hornung und M. Rabinovich, „Least-Squares Rigid Motion Using SVD“, Department of Computer Science, ETH Zurich, Jan. 2017. Adresse: <https://igl.ethz.ch/projects/ARAP/>.
- [10] J. Procházková und D. Martišek, „Notes on Iterative Closest Point Algorithm“, in *17th Conference on Applied Mathematics (APLIMAT)*, Apr. 2018, S. 876–884.
- [11] A. Fitzgibbon, „Robust Registration of 2D and 3D Point Sets“, *Image and Vision Computing*, Nr. 21, S. 1145–1153, Apr. 2002. DOI: 10.1016/j.imavis.2003.09.004.
- [12] P. J. Huber, „Robust Estimation of a Location Parameter“, *Annals of Mathematical Statistics*, Jg. 35, Nr. 1, S. 73–101, März 1964. DOI: 10.1214/aoms/1177703732.

- [13] S. Fantoni et al., „Accurate and Automatic Alignment of Range Surfaces“, in *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission*, Okt. 2012, S. 73–80. DOI: 10.1109/3DIMPVT.2012.63.
- [14] A. V. Segal et al., „Generalized-ICP“, in *Proc. of Robotics: Science and Systems*, Seattle, Juni 2009. DOI: 10.15607/RSS.2009.V.021.
- [15] M. Korn et al., „Color supported generalized-ICP“, in *2014 International Conference on Computer Vision Theory and Applications (VISAPP)*, Bd. 3, Jan. 2014, S. 592–599.
- [16] M. Miknis et al., „Near real-time point cloud processing using the PCL“, in *International Conference on Systems, Signals and Image Processing (IWSSIP)*, IEEE, Sep. 2015, S. 153–156. DOI: 10.1109/IWSSIP.2015.7314200.
- [17] D. Holz et al., „Registration with the Point Cloud Library: A Modular Framework for Aligning in 3-D“, *IEEE Robotics Automation Magazine*, Jg. 22, Nr. 4, S. 110–124, Dez. 2015, ISSN: 1558-223X. DOI: 10.1109/MRA.2015.2432331.
- [18] X.-F. Han et al., „A review of algorithms for filtering the 3D point cloud“, *Signal Processing Image Communication*, Jg. 57, Nr. 1, S. 103–112, Mai 2017. DOI: 10.1016/j.image.2017.05.009.
- [19] C. Stucker, „Semantic Point Cloud Filtering“, Sep. 2017, Master Thesis, ETH Zürich.
- [20] S. Hesse, „Optimierung von PMD-basierten Tiefenkarten mittels Moving-Least-Squares-Verfahren“, Jan. 2012, Masterarbeit, DE.
- [21] S. Yong-hua et al., „Denoising Algorithm of Airborne LIDAR Point Cloud Based on 3D Grid“, *International Journal of Signal Processing, Image Processing and Pattern Recognition*, Jg. 10, Nr. 3, S. 85–92, 2017. DOI: <http://dx.doi.org/10.14257/ijsp.2017.10.3.09>.
- [22] (Jan. 2019). Intel RealSense D400Series Product Family. Rev. 5, Intel, Adresse: <https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf> (besucht am 05.12.2019).
- [23] A. G. Jepsen et al., „Using the Intel® RealSense™ Depth cameras D4xx in Multi-Camera Configurations“, Techn. Ber.
- [24] M. Quigley et al., „ROS: an open-source Robot Operating System“, in *ICRA Workshop on Open Source Software*, Bd. 3, Jan. 2009, S. 5.

- 
- [25] R. B. Rusu und S. Cousins, „3D is here: Point Cloud Library (PCL)“, in *2011 IEEE International Conference on Robotics and Automation*, Mai 2011, S. 1–4. DOI: 10.1109/ICRA.2011.5980567.
- [26] (Juni 2019). Projection in Intel RealSense SDK 2.0, Intel, Adresse: <https://dev.intelrealsense.com/docs/projection-in-intel-realsense-sdk-20>.