Fachhochschule Westküste
Hochschule für Wirtschaft & Technik

HAW HAMBURG

Masterarbeit

Thore Maximilian Kopper

Entwicklung eines handgeführten 3D-Scanners zur Objektreproduktion mittels texturierter 3D-Punktwolken

Fachhochschule Westküste
Fachbereich Technik

Fachhochschule Westküste - University of Applied Sciences
Faculty of Engineering

Hochschule für Angewandte Wissenschaften Hamburg
Fakultät Technik und Informatik
Department Informations- und Elektrotechnik

Hamburg University of Applied Sciences
Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

Thore Maximilian Kopper

Entwicklung eines handgeführten 3D-Scanners zur Objektreproduktion mittels texturierter 3D-Punktwolken

Masterarbeit eingereicht im Rahmen der Masterprüfung
im gemeinsamen Masterstudiengang Mikroelektronische Systeme
am Fachbereich Technik
der Fachhochschule Westküste
und
am Department Informations- und Elektrotechnik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Marc Hensel
Zweitgutachter: Prof. Dr. Stephan Hußmann

Eingereicht am: 17. Dezember 2024

Thore Maximilian Kopper

Thema der Arbeit

Entwicklung eines handgeführten 3D-Scanners zur Objektreproduktion mittels texturierter 3D-Punktwolken

Stichworte

3D-Scanning, Stereokamera, ArUco-Marker, OpenCV, Open3D, Punktwolken, Punktwolkenregistrierung, ICP-Algorithmus, Oberflächenrekonstruktion

Kurzzusammenfassung

In dieser Masterarbeit wird ein System zur Erstellung von 3D-Modellen aus Aufnahmen einer mobilen Sensoreinheit entwickelt. Die erfassten Bilddaten durchlaufen eine Verarbeitungskette, die letztendlich ein vollständiges 3D-Modell erzeugt. Dabei steht ein umfassendes Gesamtsystem im Vordergrund, das sämtliche Arbeitsschritte von der Bilderfassung über die Vorverarbeitung bis zur finalen Modellierung integriert.

Thore Maximilian Kopper

Title of Thesis

Development of a Handheld 3D Scanner for Object Reproduction Using Textured 3D Point Clouds

Keywords

3D Scanning, Stereo Camera, ArUco Markers, OpenCV, Open3D, Point Clouds, Point Cloud Registration, ICP Algorithm, Surface Reconstruction

Abstract

In this thesis, a system for creating 3D models from images captured by a mobile sensor unit is developed. The acquired image data are processed through a pipeline that ultimately generates a complete 3D model. The focus is on a comprehensive integrated system that encompasses all workflow steps, from image acquisition and preprocessing to final modeling.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungen

BPA Ball pivoting Algorithmus.

CAD Computer Aided Design.

DIY Do it yourself.

FPFH Fast Point Feature Histogram.

GPLv3 GNU General Public License Version 3.

GUI Graphical User Interface.

HAW Hochschule für Angewandte Wissenschaften.

ICP Iterative closest point.

IMU Inertial Measurement Unit.

K-d-Baum K-dimensionaler Baum.

k-NN k-Nächste-Nachbarn.

ORB Oriented FAST and Rotated BRIEF.

RANSAC Random Sample Consensus.

RMS Root Mean Square.

SDK Software Development Kit.

SHOT Signature of Histograms of Orientations.

SVD Singulärwertzerlegung/Singular value decomposition.

UI User Interface.

1 Einführung

In diesem Kapitel wird das Thema der Arbeit eingeführt. Zunächst werden die Motivation für die Themenwahl und die potenziellen Anwendungsgebiete der Ergebnisse erläutert. Anschließend wird das Ziel dieser Arbeit definiert. Abschließend wird ein Überblick über den Aufbau der vorliegenden Arbeit gegeben.

1.1 Motivation

Mit einem 3D-Scanner kann die Oberfläche eines realen Objekts erfasst und daraus ein digitales Abbild erstellt werden. Heutzutage gibt es viele Anwendungsfälle für 3D-Scanner. In der Industrie sind die Vorteile schon seit Jahren bekannt. Durch die stetige Verbesserung der Genauigkeit solcher Systeme können beispielsweise maßgenaue Scans von Bauteilen zur Qualitätssicherung erstellt werden. Auch in der Humanmedizin wurden in den letzten Jahren immer mehr Anwendungsfälle entdeckt, die von der Nutzung von 3D-Scannern profitieren. Diese reichen von der Anfertigung von Prothesen bis hin zu experimentellen Behandlungsmethoden.
Während 3D-Scanner im professionellen Umfeld schon seit einigen Jahren erfolgreich und in großem Umfang eingesetzt werden, sind sie im Alltag aufgrund der recht hohen Kosten bisher wenig verbreitet, obwohl die Anwendungsfälle offensichtlich sind. Gerade in der heutigen Zeit, in der Geräte immer seltener repariert und stattdessen neu angeschafft werden, kann ein 3D-Scanner in Kombination mit einem 3D-Drucker Abhilfe schaffen. 3D-Drucker ermöglichen die präzise Reproduktion digital am Computer erstellter Modelle. Diese Modelle zu erstellen, ist eine Fähigkeit, die erlernt werden muss und nicht von jedem beherrscht wird. Gerade bei komplizierten Geometrien kann die Erstellung eines 3D-Modells selbst für erfahrene Konstrukteure ein langer Prozess sein. Ein erschwinglicher 3D-Scanner könnte hier die Brücke schlagen, indem er ermöglicht, Objekte wie Ersatzteile ohne die umständliche Erstellung von 3D-Modellen direkt zu reproduzieren.

1.2 Umfeld

Neben der offensichtlichen Möglichkeit, Objekte mithilfe von 3D-Scannern zu reproduzieren, gibt es einen weiteren spezifischen Anwendungsfall im Bereich der Luftfahrt. Bei der Lufthansa Technik AG werden sämtliche Flugzeuggeräte regelmäßig geprüft und gewartet. Im Rahmen von Funktionsprüfungen werden Cockpitpanels, die mit Schaltern, Tastern und Anzeigen ausgestattet sind, automatisiert getestet. Diese Panels sind wesentliche Bestandteile im Cockpit und erfordern eine präzise Überprüfung, um die Funktionsfähigkeit der Eingabegeräte und die Mindestleuchtstärke der Anzeigen zu gewährleisten. Derzeit wird ein Prozess implementiert, bei dem die Geräte automatisiert von einem Roboterarm getestet werden. Der Vorgang, bei dem ein bisher noch nicht verwendetes Panel manuell eingelernt wird, nimmt viel Zeit in Anspruch.
Um diesen zeitaufwendigen Prozess zu optimieren, könnte eine Methode entwickelt werden, bei der das Cockpitpanel mit einem 3D-Scanner erfasst wird, während gleichzeitig die Textur aufgenommen wird. Die 3D-Daten ermöglichen die exakte Positionierung des Roboterarms, während die Texturinformationen zur sicheren Erkennung der Eingabegeräte sowie zur Analyse von Beschriftungen und anderen Oberflächeneigenschaften dienen. Diese Verknüpfung von räumlichen und visuellen Daten bietet mit einer Datenbank für Schalter und anderen Elementen eine Basis für eine automatisierte Erstellung des Roboterprogramms, bei dem der Anwender lediglich mitteilen muss, welche Eingabegeräte wie getestet werden sollen.
So kann der Roboterarm sicher und effizient agieren, ohne die empfindlichen und teuren Eingabegeräte durch Fehlpositionierungen zu beschädigen. Ein solcher Ansatz würde nicht nur die Effizienz des Prüfprozesses deutlich steigern, sondern auch die Lebensdauer der teuren Komponenten verlängern. Durch den präzisen Einsatz der 3D-Scantechnologie könnten somit sowohl die Zuverlässigkeit der Tests als auch die Kosteneffizienz verbessert werden.

In einer vorangegangenen Abschlussarbeit hat Friederike Gollor ein System für Oberflächenscans und die Erstellung von 3D-Modellen von Personen entwickelt [41]. Mit diesem System wurden erfolgreich Scans des Kopf- und Schulterbereichs von Personen aufgenommen. Die Punktwolken wurden mit einer automatisiert im Kreis um den Kopf verfahrenden Tiefenkamera erfasst. Unter Verwendung des weit verbreiteten Iterative Closest Point (ICP) Algorithmus wurden diese Punktwolken registriert und zu einem Gesamtobjekt zusammengefügt. Anschließend wurde die Oberfläche mit dem Ball Pivoting Algorithmus rekonstruiert.

Die von Frau Gollor genutzten Verfahren und Erkenntnisse werden in dieser Arbeit zum Teil genutzt sowie optimiert und weiterentwickelt.

1.3 Zielsetzung

Das vorrangige Ziel dieser Arbeit ist es, einen funktionsfähigen Prototypen zu entwickeln, der das Scannen von Objekten ermöglicht und ein vollständiges, texturiertes 3D-Modell erzeugt. Der Fokus liegt dabei auf der händischen Erfassung von Objekten. Im Gegensatz zu der geführten Erfassung mittels Roboterarm oder einer Kamerahalterung, die sich auf einem definierten Pfad bewegt, wie es bei Frau Gollor der Fall war, erfordert die handgeführte Aufnahme andere Methodiken, mit denen die Daten aufgenommen werden können und die Punkwolkenregistrierung ermöglichen.
Im Bezug auf die automatisierten Gerätetests, soll nur die Basis in Form eines texturierten 3D-Modells gelegt werden. Ein vollständiges Ablaufprogramm zur Ansteuerung des Roboterarms soll nicht angefertigt werden. Im folgenden wird die Zielsetzung mit Hilfe der SMART-Methode konkret definiert.

Specific: Das Ziel dieser Arbeit ist die Entwicklung eines handgeführten 3D-Scanners, der in der Lage ist, präzise texturierte 3D-Punktwolken von Objekten zu erfassen. Diese Daten sollen sowohl für die Reproduktion von Objekten mit 3D-Druckern als auch für die Durchführung automatisierter Gerätetests, beispielsweise an Cockpitpanels, genutzt werden können. Der Scanner soll mobil und benutzerfreundlich sein und eine hohe Genauigkeit und Detailtreue bieten, um die erfassten Objekte realitätsnah in digitalen Modellen abzubilden.

Measurable: Die Qualität des Systems wird durch den Vergleich des erzeugten 3D-Modells mit einem maßstabsgetreuen Modell des gescannten Objekts gemessen. Dabei wird die Abweichung in den Dimensionen ermittelt. Der Erfolg wird anhand festgelegter Toleranzwerte bewertet, welche während der Anforderungsanalyse erarbeitet werden.

Achievable: Die Umsetzung verspricht eine deutliche Steigerung der Effizienz sowohl bei der Reproduktion von Objekten als auch bei der Durchführung von automatisierten Tests an Luftfahrtgeräten.

Realistic: Die Lösung wird basierend auf bereits existierender, bewährter Hardware, wie handelsüblichen 3D-Sensoren entwickelt, die für den geplanten Anwendungsfall angepasst werden. Die Softwareentwicklung greift auf etablierte Open-Source-Bibliotheken,

wie zurück. Diese werden durch spezifische, eigens entwickelte Algorithmen ergänzt, um die Anforderungen der handgeführten 3D-Datenerfassung zu erfüllen. Durch die Kombination von verfügbarer Technik und individueller Anpassung bleibt das Projekt realistisch und umsetzbar. Zusätzlich kann sich bei dem Thema auf vorangegangene Arbeiten bezogen werden.

Time-bound: Die Entwicklung soll innerhalb des Zeitrahmens der Masterarbeit, das heißt innerhalb von sechs Monaten, umgesetzt werden.

1.4 Gliederung

Nach dieser Einführung werden in Kapitel 2 die theoretischen Grundlagen zum Verständnis dieser Arbeit dargelegt. Daraufhin beleuchtet Kapitel 3 den aktuellen Stand der Technik im Bereich der 3D-Scanner, wobei sowohl bestehende Produkte als auch neueste Entwicklungsergebnisse vorgestellt werden. In Kapitel 4 werden die Anforderungen an das zu entwickelnde System erarbeitet, basierend auf den zuvor genannten Kapiteln sowie den Anforderungen der involvierten Personen und Anwendungsfälle. Kapitel 5 stellt das Konzept vor, welches als Grundlage für die Systementwicklung dient. Die eigentliche Umsetzung des Konzepts und die technische Realisierung werden in Kapitel 6 beschrieben. In Kapitel 7 wird das fertiggestellte System getestet, um zu prüfen, ob es den definierten Anforderungen gerecht wird und das Ziel erreicht wurde. Abschließend zieht Kapitel 8 ein Fazit und gibt einen Ausblick auf potenzielle Weiterentwicklungen des Systems.

2 Grundlagen

In diesem Kapitel werden die grundlegenden Konzepte der 3D-Bildgebung vermittelt, die für das Verständnis der nachfolgenden Abschnitte von zentraler Bedeutung sind. Zunächst erfolgt ein umfassender Überblick über die 3D-Messtechnik, in dem verschiedene Technologien und Methoden zur Erfassung von 3D-Daten vorgestellt werden. Hierbei werden sowohl die technischen Prinzipien als auch die praktischen Anwendungsmöglichkeiten dieser Verfahren eingehend beleuchtet. Im anschließenden Abschnitt über Punktwolken wird erläutert, wie Punktwolken strukturiert sind, welche Verfahren zur Verarbeitung dieser Daten zur Verfügung stehen und welche Methoden zur Registrierung eingesetzt werden können.

2.1 3D-Messtechnik

In der 3D-Messtechnik lassen sich grundsätzlich zwei Hauptmethoden unterscheiden: taktile und optische Verfahren. Bei der Erfassung eines dreidimensionalen Körpers wird das Objekt innerhalb eines dreidimensionalen Koordinatensystems abgebildet, wobei einzelne Punkte des Objekts in Bezug auf einen definierten Ursprung erfasst werden. Diese Punkte werden im Koordinatensystem lokalisiert und bilden zusammen das konstruierte 3D-Modell [38, 30].
Eine der grundlegendsten Methoden besteht darin, das Objekt vollständig mit einem mechanischen Tastkopf abzutasten und die erfassten Punkte zusammenzufassen. Üblicherweise wird der Tastkopf von einem Roboterarm geführt, dessen sensorbasierte Genauigkeit nur schwer zu übertreffen ist. Diese Methode ist seit vielen Jahren etabliert und kontinuierlich weiterentwickelt worden. Die Hauptvorteile der taktilen Messtechnik liegen in ihrer hohen Genauigkeit und Flexibilität, da sie in der Lage ist, auch Hinterschneidungen und komplexe Tiefenstrukturen präzise zu vermessen. Die Genauigkeit der Messung kann nahezu beliebig erhöht werden, indem der Abstand zwischen den

Messpunkten verringert wird. Jedoch sind taktile Messverfahren mit einigen wesentlichen Nachteilen verbunden. Zum einen sind die Kosten in der Regel hoch, und zum anderen gestaltet sich die Aufnahmedauer oft langwierig. Für eine präzise Abtastung ist eine große Anzahl von Messpunkten erforderlich, was den gesamten Prozess zeitintensiv macht. Darüber hinaus stoßen taktile Messverfahren an ihre Grenzen, wenn es um die Erfassung leicht verformbarer, poröser, zerbrechlicher oder elastischer Objekte geht, da diese Materialien häufig durch den Kontakt mit dem Tastkopf beeinflusst oder beschädigt werden können [94].

Die optische 3D-Messtechnik bietet einen gegensätzlichen Ansatz zur taktilen Methode. Ihr entscheidender Vorteil liegt darin, dass ein aufgenommenes Bild – je nach Qualität und Auflösung – tausende Messpunkte mit den entsprechenden Raumkoordinaten enthalten kann [38, 30]. Die Genauigkeit variiert dabei je nach angewandter Technologie. Während einige Verfahren das Innere des Objekts erfassen können, konzentriert sich diese Arbeit ausschließlich auf optische Messverfahren, die die Oberfläche des Objekts abtasten.
In Abbildung 2.1 wird eine Übersicht der Methoden zur optischen 3D-Erfassung präsentiert, wie sie von Beyerer [13] eingeteilt werden. Die dunkel hervorgehobenen Methoden werden aufgrund ihrer weiten Verbreitung in der Industrie gesondert beschrieben.

Der Grundaufbau des Messsystems sieht eine Struktur mit Sender und Empfänger vor. Es gibt sowohl aktive Systeme, bei denen ein Signal von einem Sender ausgesendet wird, als auch passive Systeme, bei denen die Daten durch Reflexionen des Umgebungslichts erzeugt werden. Zusätzlich zur Methodik unterscheidet man bei der Datenerfassung nach der Ausrichtung von Sender und Empfänger. Bei der koaxialen Antastung (in Abbildung 2.1 in Blau dargestellt) verlaufen die optischen Wege von Sender und Empfänger auf derselben Achse, was einem Winkel von 0° entspricht (siehe Abbildung 2.2a). Im Gegensatz dazu schließen Sender und Empfänger bei der Triangulation (in Abbildung 2.2b dargestellt) einen Winkel γ größer als 0° ein. Die koaxiale Antastung eignet sich gut zur Vermessung tiefer Strukturen wie Bohrungen, da es dabei zu keiner Abschattung zwischen Sender und Empfänger kommt. Ein Nachteil ist jedoch, dass diese Methode einzelne Punkte misst, was bedeutet, dass viele Einzelmessungen anschließend zu einem Gesamtobjekt zusammengefügt werden müssen. Bei der Triangulation können dagegen in der Regel mehrere Messpunkte gleichzeitig erfasst werden [13].

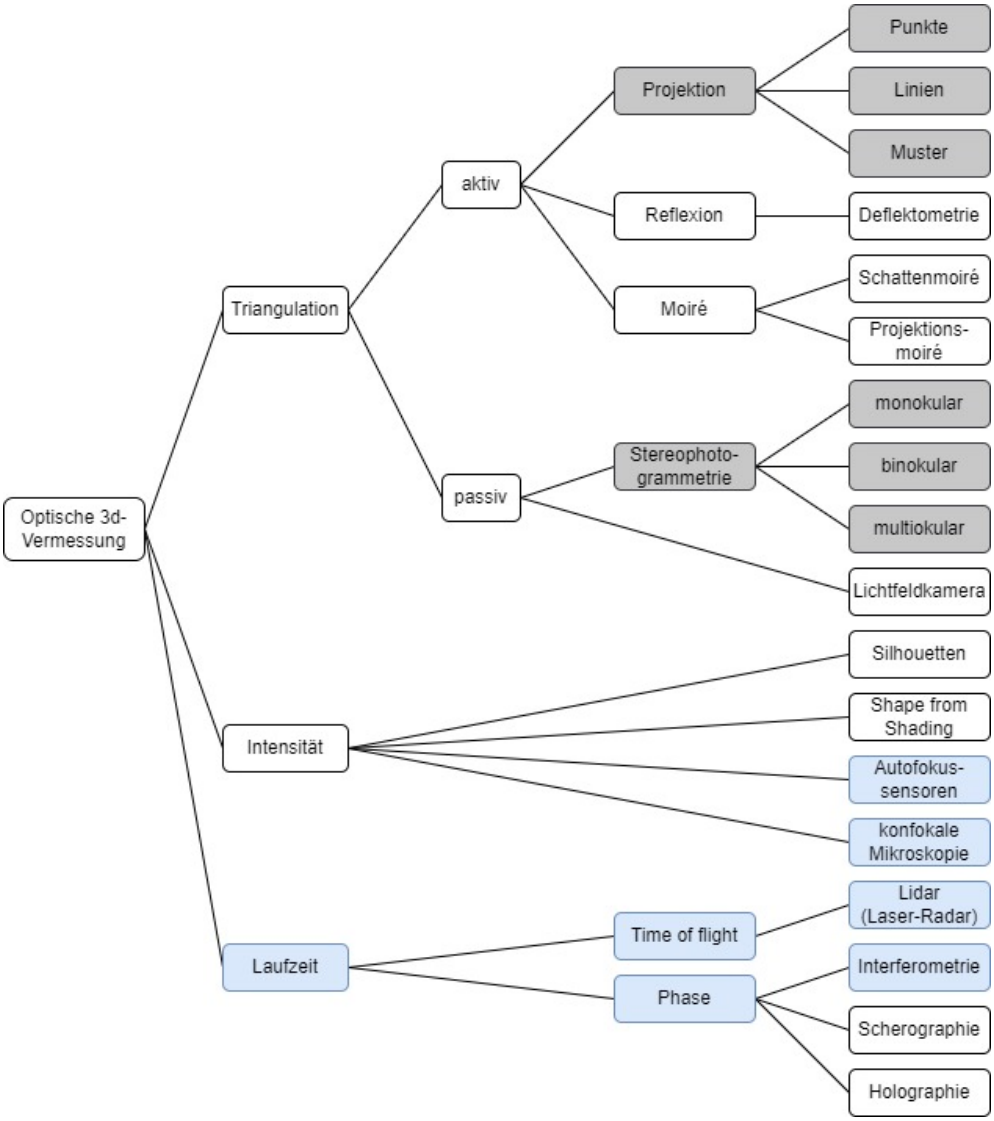
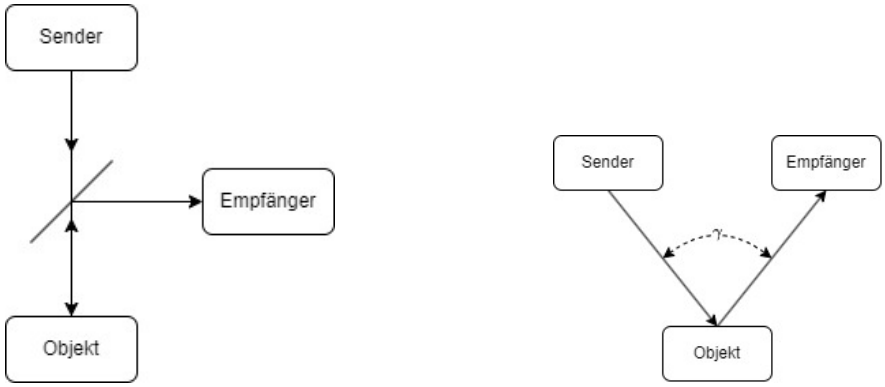


Abbildung 2.1: Optische Messverfahren nach Beyerer. Blau: Methoden, die koaxiale Antastung verwenden. Grau: Hier verwendete bzw. näher beschriebene Verfahren

2.1.1 Auswirkungen von Streuung und Reflexion

Um ein Objekt optisch zu untersuchen oder zu vermessen, muss das einfallende Licht in Richtung der Kamera reflektiert werden. Neben der Reflexion gemäß dem Reflexionsgesetz gibt es bei realen Objekten verschiedene Arten der Reflexion oder Streuung. Während die makroskopische Struktur eine bestimmte Art der Reflexion nahelegt, können



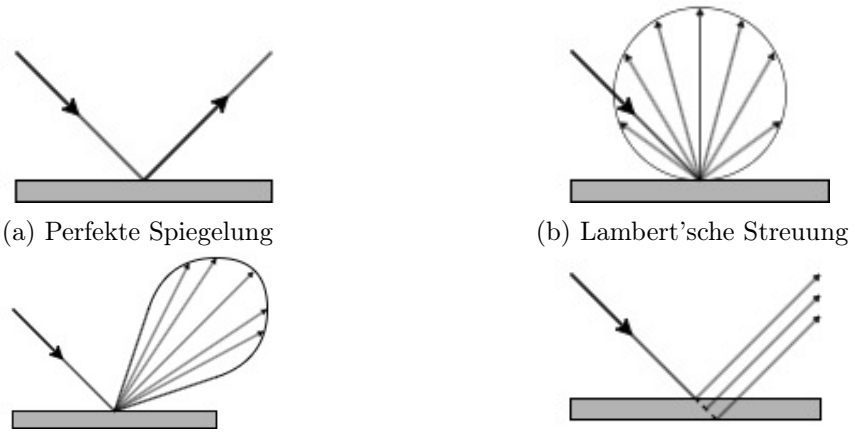
(a) Koaxiale Antastung (b) Triangulation

Abbildung 2.2: Grundanordnung nach Winkel der Sichtachsen zwischen Sende- und Empfangseinheit

mikroskopische Unebenheiten in der Größenordnung der Lichtwellenlänge dazu führen, dass Licht in unerwartete Richtungen reflektiert wird.
Teiltransparente Oberflächen können ebenfalls zu unerwarteten Ergebnissen führen. Während das Licht mit einer gewissen Wahrscheinlichkeit an der Oberfläche reflektiert wird, kommt es im Inneren des Materials zur Volumenstreuung, bei der das Licht in verschiedene Richtungen gestreut wird. Dadurch ist das Reflexionsverhalten unvorhersehbar und das Licht kann entgegen dem Reflexionsgesetz in einem völlig anderen Winkel reflektiert werden als erwartet [13].

In Abbildung 2.3 ist dargestellt, wie sich das Licht bei der Reflexion an der Oberfläche eines realen Objekts verhalten kann. In Abbildung 2.3a ist dargestellt, wie sich das Objekt bei einer perfekt spiegelnden Oberfläche verhält. Abbildung 2.3b zeigt einen Lambert'schen Strahler, bei dem Licht in jede Richtung mit der gleichen Dichte reflektiert wird. Dadurch erscheint das Objekt aus jeder Richtung gleich Hell. In der Realität tritt in der Regel ein Mischverhalten der beiden Fälle auf. In Abbildung 2.3c ist ein solches Verhalten in Form der schwachen spiegelnden Reflexion dargestellt. Das einfallende Licht wird gemäß des Reflexionsgesetzes in eine bevorzugte Richtung reflektiert, wobei das Licht zu einem gewissen Maße aufgefächert wird. Abbildung 2.3d zeigt den relevanten Strahlenverlauf am Beispiel der Volumenstreuung [13].

Bei der Anwendung in der 3D-Bildgebung müssen die Materialeigenschaften und die Beleuchtung bzw. die Senderstrahlung berücksichtigt werden. Teiltransparente Oberflächen können dazu führen, dass der Oberflächenpunkt aufgeweitet erscheint. Ein einzelner Lichtstrahl wird an verschiedenen Punkten auf und unter der Oberfläche reflektiert,



(a) Perfekte Spiegelung (b) Lambert'sche Streuung

(c) Schwach spiegelnde Reflexion (d) Volumenstreuung

Abbildung 2.3: Reflexion- und Streuungsarten

was das punktförmige Signal verbreitert und die genaue Position des Objekts im Raum beeinträchtigen kann.
Es ist wichtig sicherzustellen, dass ausreichend Licht am Sensor ankommt. Das Objekt sollte weder zu stark reflektieren, was dazu führen kann, dass Informationen den Sensor nicht erreichen, noch zu stark streuen, was zu unzureichender Lichtmenge am Sensor führen kann. Unter bestimmten Geometrien kann eine reflektierende Oberfläche auch Mehrfachreflexionen erzeugen, was zu einer Erkennung mehrerer Punkte anstelle eines einzelnen führen kann. Im Falle von Reflexionsproblemen kann die Verwendung von Mattierungssprays helfen, um eine gleichmäßige, optimal reflektierende Oberfläche zu erzielen.

2.1.2 Stereophotogrammetrie

Die Stereophotogrammetrie ist eine bewährte Methode der 3D-Bildgebung, bei der die Raumkoordinaten eines Objekts durch die Triangulation von Bildinformationen, die von zwei Kameras erfasst werden, ermittelt werden. Im Wesentlichen nutzt diese Technik zwei Kameras, deren Bildebenen im besten Fall komplanar sind, was bedeutet, dass sie in einer Ebene ausgerichtet sind. Dieses geometrische Arrangement ermöglicht es, die Unterschiede in den Bildern der beiden Kameras mit dem geringstmöglichen Aufwand zu analysieren und daraus die räumlichen Informationen des Objekts zu berechnen. In Abbildung 2.4 wird das Verfahren aus einer Draufsicht veranschaulicht. Im weiteren

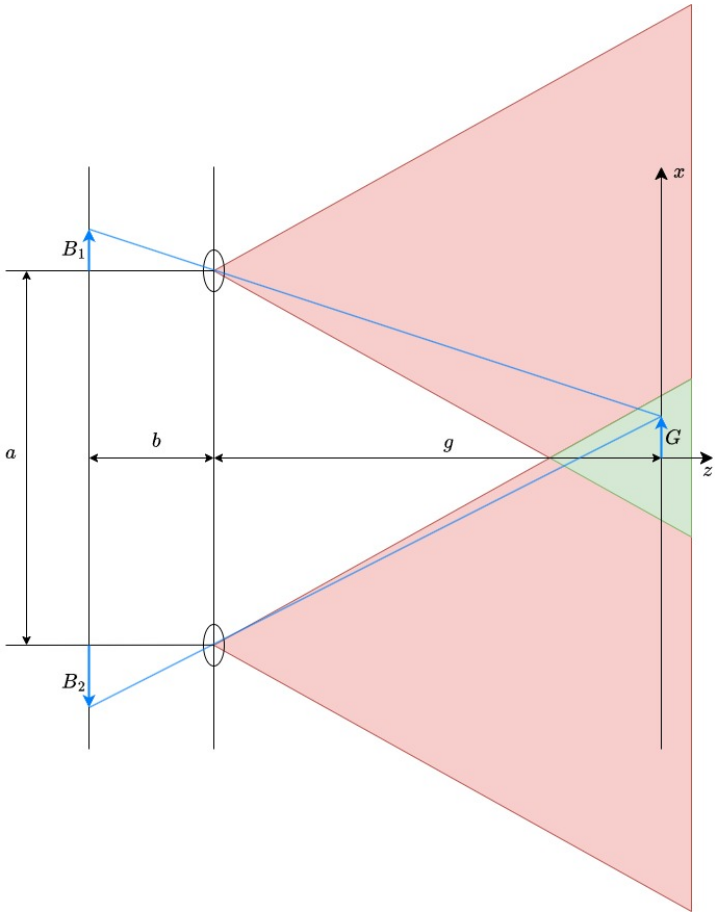


Abbildung 2.4: Prinzip der Stereophotogrammetrie

Verlauf wird das Prinzip dieses Verfahrens basiert auf basierend auf den Erklärungen von [13, 34] detailliert erläutert.

Um das Verfahren anzuwenden, muss die Bedingung erfüllt sein, dass der zu beobachtende Punkt im Sichtfeld beider Kameras vorhanden ist. In der Abbildung sind die Sichtfelder beider Kameras mit roten Kegeln dargestellt. Das zu erkennende Objekt muss sich im grün hinterlegten Bereich befinden, in dem sich die beiden Sichtkegel überschneiden. Zusätzlich muss die Lagebeziehung der beiden Kameras zueinander bekannt sein. Diese ergibt sich etwa durch eine extrinsische Kalibrierung, bei der die 3D-Position und die Ausrichtung beider Kameras mithilfe eines Prüfkörpers bestimmt werden. Im vorliegenden Fall liegen die Sichtachsen der Kameras exakt parallel (Komplanarität), und der Abstand zwischen den optischen Zentren wird als a bezeichnet. Es werden gleichartige Kameras mit einer Brennweite b verwendet. Des Weiteren definiert G die Verschie-

bung des beobachteten Punkts entlang der x-Achse. Mit g wird die Gegenstandsweite beschrieben, welche als der Abstand des Punkts zum Zentrum der Kameraanordnung definiert ist. Die Parallaxe, die Verschiebung der Position des Punkts auf den Kamerasensoren, ist als $p = B_1 - B_2$ definiert. Mithilfe des Strahlensatzes ergeben sich folgende Gleichungen, die das System beschreiben:

$$\frac{B_1}{b} = \frac{\frac{a}{2} - G}{g} \tag{2.1}$$

$$\frac{-B_2}{b} = \frac{\frac{a}{2} + G}{g} \tag{2.2}$$

Durch Umstellen und Einsetzen erhält man die folgende Formel, mit der der Abstand des Punktes zum Zentrum der Kamera ermittelt werden kann. Eine ausführliche Herleitung der Formel ist im Anhang dargestellt.

$$g = \frac{a \cdot b}{p} \tag{2.3}$$

Mit dem Abstand und dem nun bestimmbaren Schnittpunkt der Sichtstrahlen kann anschließend die 3D-Koordinate des beobachteten Punktes ermittelt werden. Bei einem statisch beobachteten Objekt ist es unerheblich, ob das Objekt von N Kameras simultan oder von einer Kamera nacheinander aus verschiedenen Perspektiven aufgenommen wird. Zudem kann eine Aufnahme aus mehr als zwei Perspektiven die Genauigkeit der bestimmten 3D-Koordinate erhöhen.

Die Anwendung dieser Technologie erfordert, dass ein Punkt im Sichtfeld beider Kameras vorhanden ist, was bei einer fixen Kameraanordnung den minimalen Arbeitsabstand bestimmt und vom Sichtfeld der verwendeten Kameras abhängt. Dabei ist zu beachten, dass die Punktdichte bei einer konstanten Auflösung mit einem größeren Sichtfeld abnimmt. Daher ist es wichtig, ein ausgewogenes Verhältnis zwischen maximaler Punktdichte und minimalem Abstand zum Objekt zu finden.

Nun ist klar, wie die Entfernung eines Punktes mit Hilfe von zwei Kameras ermittelt werden kann. Dies führt zum Korrespondenzproblem: Ein Punkt im Bild einer Kamera muss einem Punkt im Bild der anderen Kamera zugeordnet werden. In diesem Fall,

in dem die Kameraebenen komplanar zueinander sind, liegen die korrespondierenden Punkte stets auf einer Linie in der Sensorebene. Dies kann durch die Epipolarebene erklärt werden, die durch den beobachteten Punkt und die beiden optischen Zentren der Kameras aufgespannt wird. Die Bildebenen der Kameras werden durch diese Ebene geschnitten, was zur Bildung der Epipolarlinie auf der Sensorebene führt, auf welcher sich die korrespondierenden Punkte befinden. Ein exakt horizontaler Aufbau des Systems resultiert beispielsweise in horizontalen Epipolarlinien, auf denen korrespondierende Punkte gesucht werden müssen. In der idealen Theorie muss ein Punkt daher in derselben Bildzeile beider Kameras liegen, was das Korrespondenzproblem auf eine eindimensionale Suche reduziert.

Trotz dieser Vereinfachung ist die Suche nach korrespondierenden Punkten vergleichsweise schwierig. Die Darstellung eines beobachteten Objekts variiert aufgrund der verschiedenen Perspektiven, und ein Punkt, der in einem Kamerabild vorhanden ist, kann im anderen Kamerabild durch Abschattung fehlen. Die Qualität der Korrespondenzsuche hängt daher stark von der betrachteten Szene ab. Wenn ein Objekt nur wenige Strukturen aufweist, wie etwa eine weiße Wand, kann eine exakte Zuordnung schwierig bis unmöglich sein.
Eine Möglichkeit zur Korrespondenzsuche besteht darin, sich auf Merkmale des beobachteten Objekts zu beziehen, wie zum Beispiel Ecken, Kanten oder eindeutig identifizierbare Flächen in der Szene. Um nicht auf die Struktur des Objekts angewiesen zu sein, gibt es Systeme, die Muster auf das Objekt projizieren. Diese Muster können sequentielle Streifen, Grauwertbilder oder statische Muster sein. Durch diese Methoden kann die Korrespondenzsuche stark vereinfacht oder sogar nahezu vermieden werden.

2.1.3 Lichtschnittverfahren, strukturierte Beleuchtung und Streifenprojektion

Die im Folgenden betrachteten Verfahren basieren ebenfalls auf dem Triangulationsprinzip. Diese Verfahren unterscheiden sich primär in ihrem erfassbaren Messbereich und ihrer Genauigkeit. Im Gegensatz zur Stereophotogrammetrie können diese Verfahren mit nur einem Sensor bzw. einer einzigen Kamera realisiert werden. Die Beschreibung an dieser Stelle basiert ebenfalls auf Informationen von Beyerer [13].
In der einfachsten Variante dieses Systems, dargestellt in Abbildung 2.5, werden ein Streifenlaser und eine Kamera verwendet. Der Laser wird in einem definierten Winkel θ zur Kamera positioniert. Die perspektivische Verschiebung d der Laserlinie kann von

der Kamera detektiert werden (siehe Abbildung 2.6). Mit diesen Informationen kann die Höhendifferenz h des Objekts wie folgt bestimmt werden.

$$h = \frac{d}{\tan(\theta)} \tag{2.4}$$

Der Abstand der Objektoberfläche zur Kamera kann analog zur Stereophotogrammetrie durch die Ermittlung der Position bestimmt werden, an der das reflektierte Lasersignal auf den Sensor trifft. Eine gängige Methode zur punktweisen Erfassung ist das Lichtschnittverfahren, bei dem ein Linienlaserstrahl auf das Objekt projiziert wird und die reflektierte Linie zur Bestimmung der 3D-Geometrie genutzt wird. Dieses Verfahren lässt sich erweitern, indem nicht nur eine einzelne Linien, sondern ganze Lichtfächer auf das Objekt gerichtet werden, sodass flächige Abschnitte der Objektoberfläche simultan erfasst werden können.
Eine Herausforderung bei der flächigen Abtastung ist die Zuordnung der reflektierten Lichtmuster, da die Streifen durch die Geometrie des Objekts an unterschiedlichen Positionen erscheinen können. Um Verwechslungen zu vermeiden, können inhomogene, strukturierte Lichtmuster auf das Objekt projiziert werden, die eine eindeutige Zuordnung ermöglichen. Diese Technik wird als strukturierte Beleuchtung bezeichnet. Sie erlaubt es, komlizierte Objektoberflächen präzise zu erfassen, indem global eindeutige Muster verwendet werden.
Der effektive Aufnahmebereich ist dabei abhängig von der Geometrie des Objekts. Die Projektion und die Kamera können gegebenenfalls nicht auf jeden Punkt der Objektoberfläche fokussiert werden. Dadurch ist bei einer ebenen Fläche theoretisch ein größerer Aufnahmebereich möglich als bei einer Fläche mit großen Höhendifferenzen.
Bei komplexen Objekten kann es trotz der strukturierten Beleuchtung zu Mehrdeutigkeiten bei der Zuordnung der aufgenommenen Punkte zum Beleuchtungsmuster kommen. Um dies zu vermeiden, wird häufig eine codierte Beleuchtungssequenz eingesetzt. Eine solche Projektion, beispielsweise im Graycode, ermöglicht die exakte Zuordnung eines aufgenommenen Punktes zu einem projizierten Punkt. Dieses Verfahren, bekannt als Streifenprojektion, verwendet mehrere Bilder (m verschiedene Projektionen), um durch die Auswertung der Sequenz die korrekte Zuordnung zu gewährleisten.
Die Genauigkeit dieser Verfahren kann durch feinere Projektionen und die Anpassung der Lichtquelle optimiert werden, was besonders bei der Erfassung unterschiedlicher Materialeigenschaften von Vorteil sein kann. Darüber hinaus kann der Einsatz mehre-

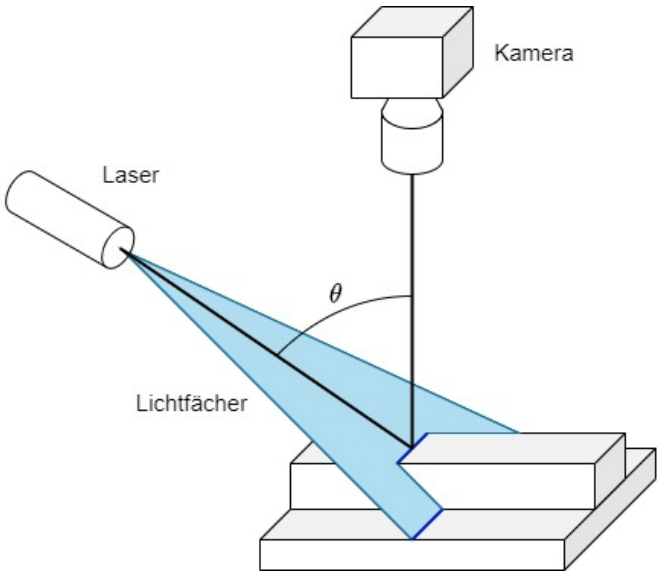


Abbildung 2.5: Prinzip Lichtschnittverfahren

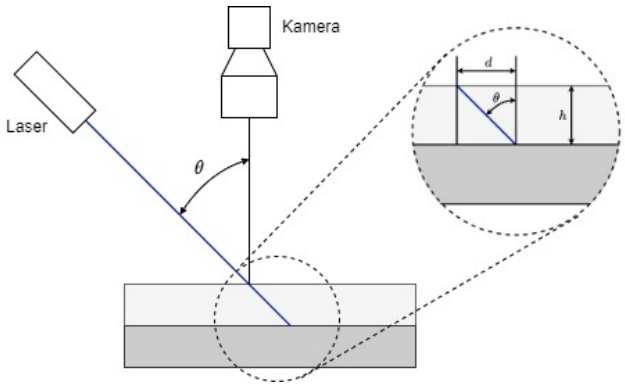


Abbildung 2.6: Berechnung Höhendifferenz

rer Kameras die Genauigkeit und Robustheit des Scans weiter verbessern, indem mehr Perspektiven und Daten zur Verknüpfung der Bildinformationen genutzt werden.

2.2 Punktwolken

Punktwolken sind eine wesentliche Datenstruktur in der 3D-Datenverarbeitung, die aus einer Vielzahl einzelner Punkte besteht, die im dreidimensionalen Raum angeordnet sind. Jeder Punkt in einer Punktwolke repräsentiert dabei eine Koordinate im Raum

und kann zusätzliche Informationen wie Farbe, Normalenvektor oder Intensität enthalten. Punktwolken werden häufig verwendet, um die Geometrie von Objekten oder Umgebungen zu beschreiben, insbesondere in Bereichen wie der Robotik, der Vermessungstechnik, der 3D-Rekonstruktion, und der Computergrafik.
Der Begriff *Wolke* spiegelt die ungeordnete Struktur einer Punktwolke wieder. Die Punkte liegen nicht zwangsläufig in einem definierten, gleichbleibenden Raster. Dieses Verhalten setzt sich in der Datenstruktur fort. Die Koordinaten sowie die weiteren Informationen der Punkte werden in Arrays gespeichert, bei denen nebeneinander liegende Informationen nicht zwangsläufig auch im 3D-Koordinatensystem nebeneinander liegen müssen [71].
Daneben können Punktwolken durch die Voxelization in ein geordnetes 3D-Raster überführt werden. Ein Voxel kann als 3D-equivalent zum 2D-pixel gesehen werden. Bei dem Vorgang wird die Punktwolke durchsucht und abhängig von der Voxelraster-Auflösung Voxel gebildet, wenn sich in diesen mindestens ein Punkt befindet. Wenn die Punkte Farbwerte oder andere zusätzliche Informationen enthalten, werden diese gemittelt für das Voxel zusammengefasst [68]. Ein Nachteil der Transformation von Punktwolken in Voxelgrids ist der Verlust von feinen Strukturen durch die Zusammenfassung mehrerer Punkte. Dieser Verlust von Informationen zieht sich durch alle Eigenschaften der Punkte durch wie zum Beispiel auch auf die Normalenvektoren [35].

Nachbarschaftsanalyse

Nachbarschaftsanalysen spielen eine zentrale Rolle in der Verarbeitung von Punktwolken. Eine Punktwolke besteht aus einer Vielzahl von diskreten 3D-Punkten, die in einem unstrukturierten Raum verteilt sind. Um Punktwolken effizient verarbeiten zu können, ist es häufig erforderlich, Informationen über die unmittelbare Nachbarschaft eines jeden Punktes zu gewinnen. Diese Informationen sind entscheidend für verschiedene Anwendungen wie die Oberflächenrekonstruktion, Glättung, Filterung oder Segmentierung von Objekten. Ein zentraler Aspekt bei der Nachbarschaftsanalyse ist dabei die schnelle und genaue Bestimmung benachbarter Punkte innerhalb der Punktwolke.
Im Rahmen der Nachbarschaftsanalyse wird für jeden Punkt in der Punktwolke nach weiteren Punkten in dessen Umgebung gesucht. Die ermittelten Nachbarpunkte liefern dabei wertvolle Erkenntnisse über die lokale Struktur und Geometrie der durch die Punktwolke dargestellten Oberfläche. In der Praxis wird je nach Anwendung häufig zwischen folgenden Nachbarschaftskriterien unterschieden [35]:

- k-Nächste-Nachbarn (k-NN): Bei diesem Ansatz wird für jeden Punkt die Gruppe der k nächstgelegenen Nachbarpunkte bestimmt. Diese Methode ist hilfreich, um lokale Eigenschaften wie die Orientierung oder die Dichte der Punktwolke in einem bestimmten Bereich zu analysieren. Durch die Wahl eines geeigneten k-Werts können sowohl detaillierte lokale Strukturen als auch globale Muster in der Punktwolke erfasst werden.

- Radius-basierte Nachbarschaft: Diese Methode ermittelt alle Punkte innerhalb eines vorgegebenen Radius um einen gegebenen Punkt. Sie eignet sich gut, um die Punktdichte einer Punktwolke zu berechnen oder zur Filterung von Ausreißern, die sich in Bereichen mit besonders geringer Punktdichte befinden.

Zusätzlich zu den genannten Nachbarschaftsdefinitionen existieren, je nach Anwendungsfall, weitere Mischformen, die Aspekte des k-NN-Ansatzes als auch der radiusbasierten Nachbarschaft kombinieren. Diese hybriden Methoden verwenden beispielsweise eine variable Nachbarschaftsgröße abhängig von der lokalen Punktdichte.
Die größte Herausforderung bei der Nachbarschaftsanalyse liegt in der Effizienz. Insbesondere bei großen Punktwolken, die aus Millionen von Punkten bestehen, wäre eine lineare Suche nach den nächsten Nachbarn für jeden Punkt extrem ineffizient und rechenaufwendig. Aufgrund der unstrukturierten Natur von Punktwolken ist ein solches Verfahren nicht praktikabel. Um dieses Problem zu lösen, kommen spezielle Datenstrukturen wie der K-d-Baum zum Einsatz.

K-d-Baum

Ein K-d-Baum ist eine Datenstruktur, die zur effiziente Organisation von Punkten in einem k-dimensionalen Raum genutzt wird. Der besondere Vorteil eines K-d-Baums liegt in der Fähigkeit, Abfragen in ungeordneten Daten, wie beispielsweise die Suche nach den nächstgelegenen Nachbarpunkten, effizient zu handhaben. Bei Punktwolken, die im 3D-Raum dargestellt werden, entspricht $k = 3$, wobei jeder Punkt durch seine x-, y- und z-Koordinate beschrieben wird. In der Theorie können weitere Dimensionen hinzugefügt werden, beispielsweise zur Suche nach den nächstgelegenen Farbwerten. Hierbei ist jedoch zu beachten, dass die Wertebereiche normalisiert werden müssen, um Abstandsberechnungen durchführen zu können. Der Einsatz eines K-d-Baums beschleunigt die Suche nach Nachbarpunkten erheblich im Vergleich zu einer linearen Suche in großen Datensätzen.

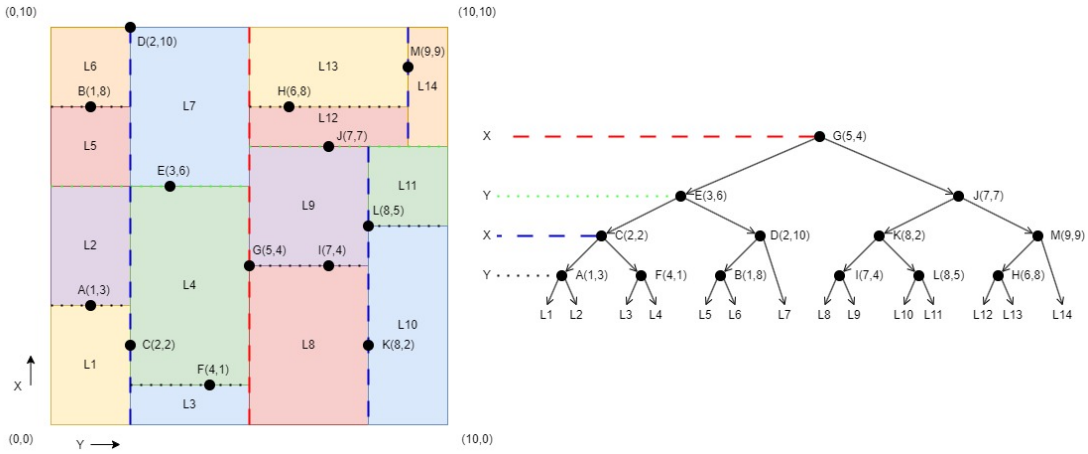


Abbildung 2.7: Darstellungsformen eines K-d-Baums

Der k-d-Baum besitzt eine binäre Baumstruktur, was bedeutet, dass jeder Knoten eine Entscheidung über die Aufteilung der Daten trifft, die zu genau zwei weiteren Knoten führt. Jeder Knoten im Baum repräsentiert eine Teilung des Merkmalsraums, wobei Punkte der linken bzw. unteren Hälfte des Baumes zugeordnet werden, wenn ihr Merkmalswert unterhalb der definierten Schwelle liegt. Liegt der Merkmalswert eines Punktes über oder gleich dieser Schwelle, wird dieser Punkt der rechten bzw. oberen Seite des Baumes zugeordnet. Die Endknoten im Baum, auch Blätter oder leaf nodes genannt, enthalten die Datenpunkte, die nicht zur Erstellung des Baums genutzt wurden. Die Tiefe des Baumes definiert, wie oft der Merkmalsraum maximal unterteilt wird, wodurch der Suchraum eingeschränkt werden kann.

Aufbau eines K-d-Baums Der Aufbau eines K-d-Baums erfolgt rekursiv, indem die Punkte schrittweise auf die Blätter des Baums verteilt werden. Dieser Prozess endet entweder, wenn die Blätter nicht weiter aufgeteilt werden können, da keine Punkte mehr vorhanden sind, oder wenn die gewünschte Tiefe des Baums erreicht ist – letzteres ist der Regelfall. Die Aufteilung der Punkte wird mit dem Ziel durchgeführt, einen gut ausbalancierten Baum zu erstellen, bei dem die Blätter eine möglichst gleiche Anzahl an Punkten enthalten. Ein solcher Baum hat den Vorteil, dass Suchanfragen gleichmäßig über den Baum verteilt werden, was zu einer konsistenten und effizienten Laufzeit führt. Der Aufbau eines K-d-Baums wird im Folgenden anhand der Koordinatendarstellung und der planaren Darstellung in Abbildung 2.7 schrittweise erläutert. In diesem Beispiel wird aus einer Menge von zweidimensionalen Punkten (A bis M) ein Baum erstellt.

- Zunächst wird ein Koordinatenraum mit 2 Dimensionen erstellt, in dem die Diskriminatoren der Punkte (X und Y) aufgetragen werden.
- Der Koordinatenraum wird entlang des ersten Diskriminators (X) geteilt. Die Teilungsgrenzen sind in den Abbildungen farbig gekennzeichnet. Um einen ausbalancierten Baum zu erzeugen, erfolgt die Teilung beim Median aller Werte des betrachteten Diskriminators. Somit befindet sich nach der Teilung auf beiden Seiten eine gleiche Anzahl an Punkten. In diesem Beispiel erfolgt die erste Teilung bei Punkt G mit $X = 5$. Punkte mit einem Diskriminatorwert kleiner als diese Schwelle werden links bzw. unterhalb der Grenze verortet, während Punkte mit einem größeren oder gleichen Wert rechts bzw. oberhalb der Grenze angeordnet werden.
- Die auf einer Seite zugeordneten Punkte werden anschließend entlang des zweiten Diskriminators (Y) geteilt. Links von G erfolgt die Teilung bei Punkt E mit $Y = 6$, rechts von G wird der Bereich bei Punkt J mit $Y = 7$ geteilt.
- Jeder neu gebildete Bereich wird dann abwechselnd entlang der beiden Diskriminatoren weiter unterteilt, bis entweder die gewünschte Tiefe des Baumes erreicht ist oder keine weiteren Punkte in einem Bereich vorhanden sind, die aufgeteilt werden könnten.
- Am Ende entstehen Regionen im Koordinatenraum, die den Blättern des Baumes (L1 bis L14) entsprechen und eine beliebige Anzahl von Punkten enthalten können. In diesem Fall wurden jedoch alle Punkte zum generieren des Baums verwendet.

In einem 3-dimensionalen K-d-Baum, der eine Punktwolke repräsentiert, erfolgt die Teilung des Koordinatenraums abwechselnd entlang der Medianwerte der X-, Y- und Z-Koordinaten. Dies ermöglicht eine effiziente Aufteilung der Daten im Raum. Wenn weitere Punkte hinzugefügt werden sollen, wird der Baum durchlaufen, wobei die neuen Punkte gemäß ihrer Koordinaten entlang der bereits vorhandenen Teilungen sortiert werden. Am Ende des Prozesses wird der neue Punkt einem der Blätter zugeordnet, ohne die bestehende Struktur des Baumes signifikant zu verändern.

Suche in einem K-d-Baum Bei der Suche nach einem Nachbarpunkt in einem K-d-Baum ermöglicht das zugrunde liegende Konzept, dass nicht jeder einzelne Punkt überprüft werden muss. Stattdessen können ganze Bereiche ausgeschlossen werden, indem einfache Vergleichsoperationen (größer/kleiner) verwendet werden, um potenzielle

Nachbarn zu identifizieren. Diese Vorgehensweise führt zu einer signifikanten Reduzierung der Anzahl der zu durchsuchenden Punkte, was insbesondere bei großen Punktwolken zu einer erheblichen Verbesserung der Recheneffizienz beiträgt. Der Suchprozess erfolgt iterativ in den folgenden Schritten, beginnend an der Wurzel des Baumes.

- Wenn der aktuelle Knoten einen geringeren Abstand als der bisher ermittelte minimale Abstand aufweist, wird dieser Punkt als nächster Punkt gespeichert, und der minimale Abstand wird entsprechend aktualisiert.

- Ein Knoten ohne Folgeknoten signalisiert, dass entweder die maximale Tiefe des Baumes erreicht wurde und die darunter liegenden Punkte einzeln überprüft werden müssen, oder, im vorliegenden Beispiel, dass alle Punkte zur Erstellung des Baumes verwendet wurden und daher keine untergeordneten Punkte existieren, was bedeutet, dass der Bereich nicht weiter überprüft werden muss (Abbruchkriterium).

- Wenn der Bereich des aktuellen Knotens, im 2-Dimensionalen durch den rechteckigen Bereich der Folgeknoten definiert, keinen Punkt enthalten kann, der näher liegt als der bisher nächstgelegene Punkt, muss dieser Zweig nicht durchsucht werden (Abbruchkriterium).

- Abschließend müssen die Zweige, die vom aktuellen Knoten abzweigen, untersucht werden. Dabei muss entschieden werden, ob zuerst der rechte oder der linke Pfad verfolgt wird. Diese Entscheidung erfolgt anhand des aktuell betrachteten Teilungsparameters, in diesem Fall X oder Y. Wenn der X-Wert betrachtet wird und der Zielpunkt einen größeren X-Wert aufweist, wird zunächst der rechte Pfad verfolgt. Es müssen beide Pfade überprüft werden, jedoch erhöht diese Vorgehensweise die Wahrscheinlichkeit, dass ein Punkt gefunden wird, der näher am Zielpunkt liegt, als der Bereich des anderen Zweiges es zulässt. Infolgedessen kann der jeweils andere Zweig möglicherweise übersprungen werden.

Um das Konzept der Nachbarschaftssuche in einem k-d-Baum zu verdeutlichen, wird im Folgenden schrittweise beschrieben, wie der nächste Nachbarpunkt des in der Abbildung eingezeichneten Punktes Q ermittelt wird [21]. Punkt Q befindet sich im Blattknoten 7. In Abbildung 2.7 ist zu erkennen, dass eine direkte Abstandsberechnung aller Punkte im Blattknoten 7 (Punkt D und E) nicht zum gewünschten Ergebnis führt, da Punkt H, welcher sich im Blattknoten 13 befindet, näher an Q ist als die beiden anderen Punkte.

- Die Suche beginnt am Wurzelknoten *G(5,4)*. Zu Beginn sind der nächstgelegene Punkt sowie der geringste Abstand noch nicht definiert. Daher wird G als der nächstgelegene Punkt festgelegt, und der euklidische Abstand zwischen den Punkten wird berechnet und gespeichert. Anschließend wird der linke Bereich von G untersucht, da der X-Wert von Q kleiner ist als der von G.

- Der nächste Knoten ist *E(3,6)*. Der Abstand zwischen Q und E ist geringer als der bisherige Abstand, weshalb der nächstgelegene Punkt und der geringste Abstand aktualisiert werden. Danach wird der Bereich oberhalb von E untersucht, da der Y-Wert von Q größer ist als der von E.

- Der Abstand von Q zu *D(2,10)* wird berechnet. Da dieser größer als der bisherige geringste Abstand ist, werden die Werte nicht aktualisiert. Dem Schema folgend würde der Bereich rechts von D untersucht werden, in dem sich aber keine weiteren Punkte befinden. Im reale Fall würde sich am Ende des Baumes in einem Blatt eine Menge an Punkten befinden die jeweils separat auf Ihren Abstand untersucht werden müssten. Stattdessen wird der Baum nun umgekehrt durchquert und die jeweils andere Seite der Schnittpunkte untersucht.

- Dem Baum nach oben folgend ist der nächste Schritt der Knoten *B(1,8)*. Dieser liegt ebenfalls weiter weg, und wird daher nicht übernommen. Die beiden Abgehenden Bereiche L5 und L6 enthalten ebenfalls keine weiteren Punkte, wodurch auch diese nicht überprüft werden müssen.

- Im nächsten Schritt wird der Knoten *C(2,2)* überprüft, der sich jedoch auch nicht näher an Q befindet. Allerdings besteht die Möglichkeit, dass sich im folgenden Bereich ein näherer Punkt verbirgt, da der kombinierte Folgebereich aus L4 und L3 innerhalb des Radius des geringsten Abstands liegt.

- Knoten *F(4,1)* erzeugt einen Bereich (L4), der zwar innerhalb des Radius des geringsten Abstands liegt, jedoch keine Folgepunkte enthält, sodass der Knoten verlassen werden kann.

- Es folgt die Untersuchung des Knotens *A(1,3)*. Dieser ist der erste Knoten, dessen Bereiche außerhalb des Radius des geringsten Abstands liegen. Daher kann dieser Knoten direkt übersprungen werden.

- Nach der vollständigen Überprüfung des linken Teils des Baumes wird nun die rechte Seite untersucht.

- Der erste Punkt auf der rechten Seite ist *J(7,7)*, der sich ebenfalls nicht näher an Q befindet. Anschließend wird der Bereich oberhalb von J untersucht, da der Y-Wert von Q größer ist als der von J.
- Knoten *M(9,9)* erfüllt die Bedingung eines kleineren Abstands ebenfalls nicht. Daher wird als Nächstes der linke Bereich von M untersucht, da der X-Wert von Q kleiner ist als der von M.
- Schließlich wird Knoten *H(6,8)* untersucht, der als neuer nächster Punkt gespeichert wird. Die Überprüfung dieses Zweiges kann anschließend abgebrochen werden, da die nachfolgenden Bereiche keine weiteren Punkte enthalten.
- Zum Abschluss müssen noch die Knoten *K(8,2)* und *I(7,4)* überprüft werden. Beide sind selbst keine nächstgelegenen Punkte, und im Fall von I sind keine weiteren Folgepunkte vorhanden.

In diesem Fall wurden lediglich die Knoten A und I nicht überprüft, was im Vergleich zur linearen Suche nur einen geringen Gewinn darstellt. Dies ist auf die geringe Anzahl der genutzten Punkte zurückzuführen. Befinden sich innerhalb des Blattknotens, in dem sich der untersuchte Punkt befindet, weitere Punkte, die näher an diesem Punkt liegen als alle anderen Bereiche, muss der Baum entsprechend seiner Tiefe vollständig von oben nach unten durchlaufen werden, wonach die Suche in der Regel abgebrochen werden kann.
Bei der Suche nach N-Nachbarpunkten wird anstelle eines einzelnen nächsten Punktes eine Menge von Punkten gespeichert. Neu gefundene Punkte werden dabei an die entsprechende Stelle im Speicher eingefügt.

2.3 Registrierung von Punktwolken

Die automatische Registrierung von Punktwolken ist von zentraler Bedeutung in der 3D-Bildgebung und bildet die Grundlage für zahlreiche Anwendungen wie 3D-Modellierung, Robotik und Umgebungsrekonstruktion. Durch die Registrierung können mehrere Punktwolken, die aus unterschiedlichen Perspektiven oder zu verschiedenen Zeitpunkten erfasst wurden, in ein gemeinsames Koordinatensystem überführt werden. Dies ermöglicht die Erstellung eines konsistenten und umfassenden 3D-Modells der erfassten Szene oder des Objekts.
Unter der Registrierung von Punktwolken versteht man den Prozess der geometrischen

Ausrichtung von zwei oder mehr Punktwolken, sodass sie optimal übereinstimmen. Das Hauptziel dieses Prozesses ist die Bestimmung der optimalen Transformation, welche die Punktwolken in Einklang bringt. Diese Transformation umfasst in der Regel Translationen und Rotationen, kann aber je nach Anwendung auch Skalierungen beinhalten. Eine präzise Registrierung ist unerlässlich, um die Genauigkeit und Integrität der resultierenden 3D-Modelle zu gewährleisten.
Registrierungsmethoden können nach verschiedenen Kriterien klassifiziert werden. Beispielsweise lassen sie sich in globale und lokale Ansätze unterteilen. Globale Methoden suchen nach der optimalen Ausrichtung im gesamten Raum ohne Vorwissen über die anfängliche Position der Punktwolken. Lokale Methoden hingegen führen eine Feinausrichtung durch, die benachbarte Punkte in Übereinstimmung bringt, und setzen oft eine grobe Vorregistrierung voraus. Eine weitere Klassifikation unterscheidet zwischen merkmalsbasierten und dichten Methoden. Merkmalsbasierte Ansätze extrahieren charakteristische Merkmale oder Schlüsselpunkte aus den Punktwolken und nutzen diese zur Bestimmung von Korrespondenzen. Dichte Methoden arbeiten direkt mit allen verfügbaren Punkten und minimieren beispielsweise den Abstand zwischen den Punktmengen insgesamt.
In der Praxis werden häufig verschiedene Registrierungsmethoden kombiniert, um sowohl Effizienz als auch Genauigkeit zu erreichen. Zunächst erfolgt eine grobe Ausrichtung mittels globaler Methoden, beispielsweise durch merkmalsbasierte Ansätze. Anschließend wird eine lokale Registrierung mithilfe von Algorithmen wie dem ICP-Algorithmus durchgeführt, um die Feinausrichtung zu optimieren. Der ICP-Algorithmus gilt als De-facto-Standard für die lokale Registrierung von Punktwolken. Allerdings erfordert er eine gute Vorregistrierung, um zu einer global optimalen Lösung zu konvergieren, und ist zudem rechenintensiv [17].

2.3.1 Globale Registrierung

Globale Registrierungsverfahren zielen darauf ab, ohne Vorwissen über die relative Position der Punktwolken eine korrekte Ausrichtung zu erzielen. Sie sind entscheidend, um eine initiale grobe Ausrichtung zu bestimmen, die als Ausgangspunkt für lokale Methoden dient. Im Folgenden werden zwei gängige Methoden zur globalen Registrierung beschrieben: merkmalsbasierte und markerbasierte Methoden.

Merkmalsbasierte Methoden

Merkmalsbasierte Methoden nutzen charakteristische Merkmale oder Schlüsselpunkte innerhalb der Punktwolken zur Registrierung. Diese Merkmale können lokale Geometrien, Kanten, Ecken oder andere signifikante Strukturen sein, die in beiden zu registrierenden Punktwolken vorhanden sind. Der Prozess der merkmalsbasierten Registrierung umfasst in der Regel die folgenden Schritte:

Merkmalsextraktion

Der erste Schritt besteht darin, markante Merkmale oder Schlüsselpunkte in jeder Punktwolke zu identifizieren. Es existieren verschiedene Algorithmen zur Extraktion von Merkmalen [93]. Viele etablierte Methoden arbeiten mit Polygonnetzen, die durch eine Oberflächenrekonstruktion aus den Punktwolken erzeugt werden. Diese Netze bestehen aus untereinander mit Kanten verbundenen Punkten, was die effiziente Berechnung von Normalenvektoren der resultierenden Flächen ermöglicht. Beispielsweise wird in [49] beschrieben, wie Kanten in Polygonnetzen durch die Analyse der Normalenvektoren identifiziert werden können.
Alternativ gibt es Methoden, die direkt auf den Punktwolken ohne vorherige Netzgenerierung operieren. In [46] wird eine solche Methode vorgestellt, bei der durch Nachbarschaftsanalysen lokale Flächen approximiert und daraus Normalenvektoren berechnet werden. Anhand dieser Normalenvektoren wird ein Maß definiert, das die Wahrscheinlichkeit angibt, ob ein Punkt zu einem Merkmal gehört. Durch weitere Analysen kann die spezifische Art des Merkmals bestimmt werden.

Merkmalsbeschreibung

Nach der Extraktion werden die Merkmale mithilfe von Deskriptoren dargestellt, die ihre geometrische oder topologische Beschaffenheit repräsentieren. Diese Deskriptoren sollen die Merkmale unabhängig von Größe oder Orientierung eindeutig beschreiben.
Ein häufig verwendeter Deskriptor in der Verarbeitung von Punktwolken ist der Signature of Histograms of Orientations (SHOT)-Deskriptor [80]. Der SHOT-Deskriptor kombiniert Informationen über die räumliche Verteilung von Normalenvektoren in der Nachbarschaft eines Punktes, um einen robusten und unverwechselbaren Merkmalsvektor zu erstellen.
Eine weitere verbreitete Methode ist der Fast Point Feature Histogram (FPFH)-Deskriptor [78]. Herkömmliche Methoden beschreiben Merkmale oft mit einem einzelnen skalaren

Wert, was dazu führen kann, dass viele Punkte in einer Szene ähnliche Beschreibungen aufweisen und somit falsche Korrespondenzen entstehen. Der FPFH-Deskriptor adressiert dieses Problem, indem er die Nachbarschaft eines Punktes anhand der geometrischen Beziehungen zwischen den Nachbarpunkten in Form eines Merkmals-Histogramms beschreibt. Dieser Ansatz wurde in späteren Arbeiten weiterentwickelt, um eine schnelle Charakterisierung der lokalen Umgebung eines Punktes zu ermöglichen [79]. Durch die reduzierte Berechnungszeit ist der FPFH-Deskriptor besonders geeignet für die Registrierung großer Punktwolken.

Merkmalskorrespondenz und Transformationsschätzung

Basierend auf den Ähnlichkeiten der Merkmalsdeskriptoren werden Korrespondenzen zwischen den Merkmalen der verschiedenen Punktwolken gesucht. Anschließend wird aus diesen Korrespondenzen eine mögliche Transformation ermittelt.
Eine häufig verwendete Methode zur Transformationsschätzung ist der Random Sample Consensus (RANSAC)-Algorithmus [32]. Dieser Algorithmus sucht iterativ nach der Transformation, die den besten Konsens unter den korrespondierenden Merkmalen liefert. Dazu wird in jeder Iteration eine zufällige Stichprobe von Merkmalspaaren ausgewählt, aus der eine Transformation berechnet wird. Bei der Anwendung auf Punktwolken sind mindestens drei Punktepaare erforderlich, um eine eindeutige Transformation zu bestimmen. Nach jeder Transformation wird bewertet, wie gut sie mit den restlichen korrespondierenden Punkten übereinstimmt. Falls die Transformation eine geringere Fehlermetrik aufweist als die bisher beste, wird sie als neue aktuelle Lösung übernommen. Dieser Prozess wird wiederholt, bis eine Abbruchbedingung erfüllt oder eine vorgegebene Anzahl von Iterationen erreicht ist. Durch die Robustheit gegenüber Ausreißern und die zufällige Auswahl von Punktkombinationen ist der RANSAC-Algorithmus besonders zuverlässig, selbst bei einer großen Anzahl von fehlerhaften Korrespondenzen.

Markerbasierte Methoden

Markerbasierte Methoden nutzen physikalische oder virtuelle Referenzpunkte (Marker), die an definierten Positionen auf oder in der Nähe des zu erfassenden Objekts platziert werden. Diese Marker dienen als visuelle oder geometrische Ankerpunkte und können in allen Punktwolken identifiziert werden, um diese zueinander auszurichten. Die Verwendung von Markern ermöglicht eine direkte und oft sehr genaue Bestimmung der

Transformation zwischen den Punktwolken. Allerdings ist diese Methode nicht anwendbar, wenn der Aufnahmebereich es nicht zulässt, dass Marker ausgelegt oder angebracht werden können. In der vorliegenden Arbeit wird die markerbasierte Methode zur globalen Registrierung eingesetzt, wie in Abschnitt 6.10 detailliert beschrieben. Der Prozess der globalen Registrierung mit Markern verläuft ähnlich zur merkmalsbasierten Methode und umfasst die folgenden Schritte:

Marker-Erkennung
Nach der Erfassung der Punktwolken müssen die Marker identifiziert werden. Dies erfolgt mithilfe spezieller Algorithmen, die entweder in den Punktwolken nach charakteristischen geometrischen Mustern suchen oder in den zugrunde liegenden 2D-Bildern nach optischen Markierungen. Ein Vorteil der Verwendung von Markern ist die Möglichkeit der eindeutigen Zuordnung, insbesondere wenn Marker mit individuellen Identifikatoren eingesetzt werden.

Schätzung der Transformation
Basierend auf den erkannten Markern wird die Transformation zwischen den Punktwolken berechnet. Wenn die Markerkoordinaten direkt aus den Punktwolken extrahiert werden, hängt die Qualität der Transformation von der Genauigkeit der Punktdatenerfassung ab. Alternativ können die Marker aus 2D-Aufnahmen extrahiert werden. In diesem Fall kann die Transformation direkt aus den Bildinformationen ermittelt werden, sofern Kalibrierinformationen über die Aufnahmehardware vorliegen. Die Genauigkeit der Transformation ist dann abhängig von der Auflösung der 2D-Daten und der Qualität der Kalibrierung. Durch die Verwendung von Markern kann oft eine hohe Registrierungsgenauigkeit erreicht werden, was besonders in Anwendungen mit strengen Genauigkeitsanforderungen von Vorteil ist.

2.3.2 Lokale Registrierung: ICP-Algorithmus

Die lokale Registrierung von Punktwolken wird im Folgenden durch die Anwendung des ICP-Algorithmus beschrieben. In der Abschlussarbeit von Friederike Gollor [41] wird im Kapitel *Grundlagen* das ICP-Verfahren in seiner ursprünglichen Form, wie es von Chen und Medioni [17] veröffentlicht wurde, detailliert erläutert. Seit der Veröffentlichung im Jahr 1992 wurden zahlreiche Varianten des ICP-Verfahrens entwickelt. Im Folgenden

wird zunächst der grundlegende Ansatz des Verfahrens vorgestellt. Anschließend werden die Modifikationen diskutiert, die in späteren Arbeiten an den einzelnen Schritten vorgenommen wurden, sowie deren Auswirkungen auf die Registrierung.

ICP-Algorithmus: Ursprüngliches Verfahren

Der ICP-Algorithmus zielt darauf ab, eine Fehlermetrik zu minimieren, die die Übereinstimmung zweier Punktwolken quantifiziert, indem eine Transformation bestimmt wird, die die Ausgangspunktwolke an die Zielpunktwolke angleicht. Jede Iteration des ursprünglichen Verfahrens folgt dabei einem festen Muster: Auswahl von Punkten, Finden von Korrespondenzen und Minimierung des Fehlers. Abbildung 2.8 illustriert dieses Verfahren anhand zweier 2D-Punktwolken.
Zunächst wird ein Satz von Punkten $\overrightarrow{\mathbf{p_i}}$ aus der Ausgangspunktwolke (oder aus beiden Punktwolken) ausgewählt, die in der aktuellen Iteration verwendet werden sollen. Da bei dieser Methode keine spezifischen geometrischen Merkmale berücksichtigt werden, können die Punkte in regelmäßigen Abständen gewählt werden, was bei großen Punktwolken den Rechenaufwand reduziert, dadurch, dass nicht jeder Punkt betrachtet wird. Anschließend werden Korrespondenzen zwischen den ausgewählten Punkten und den Punkten der Zielpunktwolke, bezeichnet als $\overrightarrow{\mathbf{q_i}}$, gesucht. Im einfachsten Fall wird jedem Punkt der Ausgangspunktwolke der nächstgelegene Punkt in der Zielpunktwolke zugeordnet.
Am Ende der Iteration wird eine Transformation $\mathbf{T}$ ermittelt, die die Fehlermetrik minimiert. Bei der Point-to-Point-Metrik ist dies der euklidische Abstand zwischen den korrespondierenden Punkten. Die Minimierungsaufgabe lautet somit:

$$\mathbf{T} = \arg\min_{\mathbf{T}} \sum_{i=1}^{N} \left\| \mathbf{T}\overrightarrow{\mathbf{p_i}} - \overrightarrow{\mathbf{q_i}} \right\|^2 \tag{2.5}$$

Eine alternative Metrik, ebenfalls von Chen und Medioni eingeführt, ist die Point-to-Plane-Metrik. Hierbei werden zunächst die Normalenvektoren $\mathbf{n}_i$ der Zielpunkte berechnet, wodurch die Tangentialebenen $\mathbf{S}_i$ an den Punkten definiert werden. Anschließend wird der Abstand l_i der transformierten Ausgangspunkte zu diesen Ebenen minimiert, was zur folgenden Minimierungsaufgabe führt:

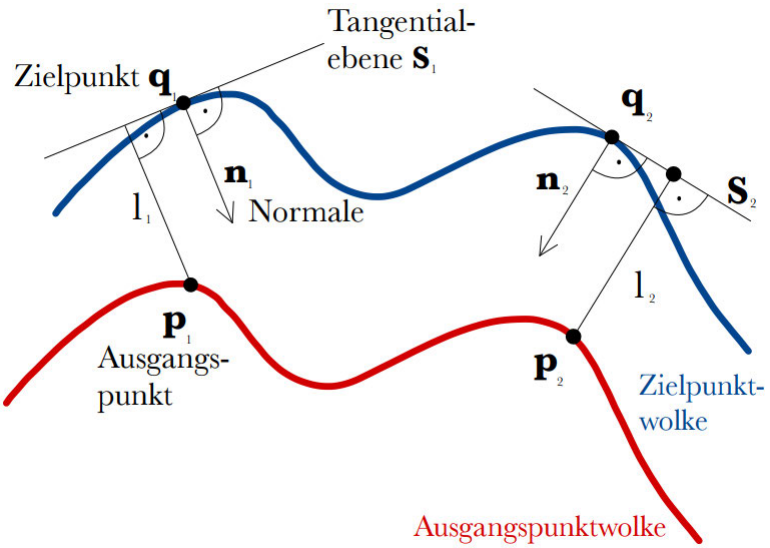


Abbildung 2.8: Illustration des ICP-Verfahrens mit Point-to-Plane-Fehlermetrik [41]

$$\mathbf{T} = \arg\min_{\mathbf{T}} \sum_{i=1}^{N} ((\mathbf{T}\vec{\mathbf{p_i}} - \vec{\mathbf{q_i}})\vec{\mathbf{n_i}})^2 \tag{2.6}$$

Die Unterschiede zwischen den beiden Fehlermetriken lassen sich anschaulich durch das Konzept von Federn verdeutlichen. Bei der Point-to-Point-Metrik sind die korrespondierenden Punktpaare durch Federn direkt miteinander verbunden. Während des Angleichungsprozesses bewegen sich diese Punkte aufeinander zu, sodass die Punktwolken idealerweise nach einigen Iterationen übereinanderliegen.
Bei der Point-to-Plane-Metrik ist die Feder hingegen auf einer Seite mit einem Punkt und auf der anderen Seite mit der Tangentialebene des korrespondierenden Punktes verbunden, entlang der sie sich frei bewegen kann. Auch hier führen mehrere Iterationen dazu, dass die Punktwolken übereinstimmen.
Ein wesentlicher Vorteil der Point-to-Plane-Metrik gegenüber der Point-to-Point-Metrik besteht darin, dass falsch zugeordnete Punktpaare die Konvergenzgeschwindigkeit weniger stark beeinflussen. Während sich die Ausgangspunktwolke in die optimale Position bewegt, können sich die virtuellen Federn entlang einer näherungsweise korrekten Tangentialebene bewegen, selbst wenn die exakte Übereinstimmung noch nicht erreicht ist. Zudem konvergiert der Algorithmus bei Verwendung der Point-to-Plane-Metrik in der

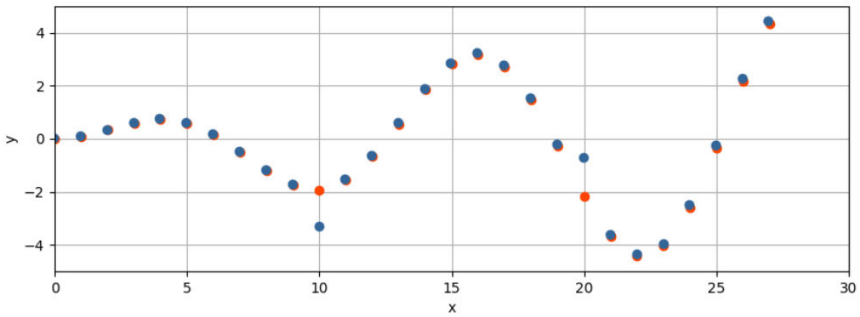


Abbildung 2.9: Registrierungsergebnis bei geringfügigen Ausreißern

Regel schneller [73].
Allerdings setzt die effektive Anwendung des ICP-Algorithmus voraus, dass die Punktwolken bereits eine grobe Vorregistrierung aufweisen. Ohne eine ausreichende anfängliche Ausrichtung können keine korrekten Korrespondenzen gefunden werden, was die Konvergenz verhindert. Zudem sollten die Punktwolken keine signifikanten Ausreißer enthalten, da diese die Registrierung negativ beeinflussen können. Abbildung 2.9 zeigt das Ergebnis einer zweidimensionalen Registrierung mit geringfügigen Ausreißern, bei der die Konvergenz zu einem guten Ergebnis führt. Dagegen ist in Abbildung 2.10 das Ergebnis bei größeren Ausreißern dargestellt, das deutlich schlechter ausfällt. Dies verdeutlicht, dass kleine Ausreißer für eine ausreichend genaue Registrierung vernachlässigt werden können, während größere Ausreißer herausgefiltert werden sollten.

ICP-Algorithmus: Weiterentwickelte Verfahren

Zehn Jahre nach der Einführung des ICP-Verfahrens haben Rusinkiewicz und Levoy [77] eine umfassende Analyse der bis dahin vorgeschlagenen Verbesserungen durchgeführt. Sie klassifizieren die Varianten des Verfahrens anhand der Schritte, in denen Modifikationen vorgenommen wurden:

- **Auswahl** eines Satzes von Punkten aus einem oder beiden Datensätzen
- **Zuordnung** der ausgewählten Punkte zu Punkten der anderen Punktwolke
- **Gewichtung** der gefundenen Korrespondenzen
- **Verwerfen** von Korrespondenzen durch Analyse der einzelnen Paare oder des gesamten Datensatzes

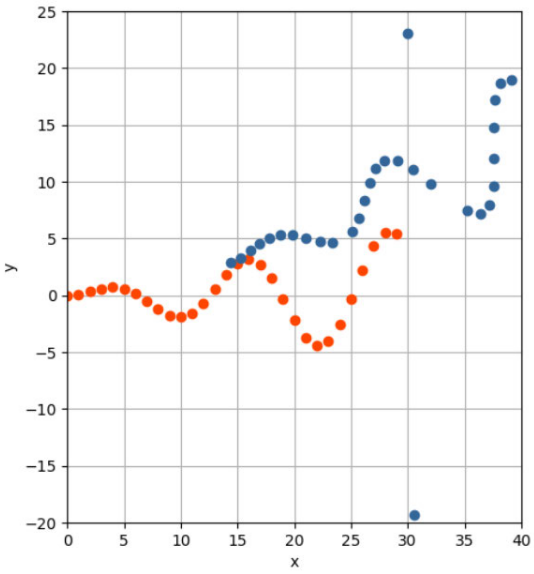
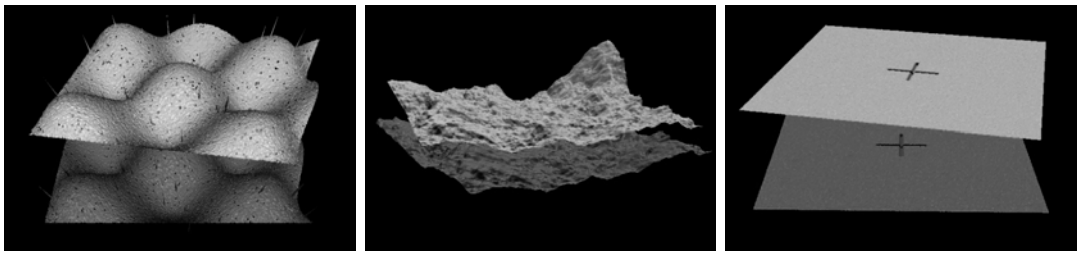


Abbildung 2.10: Registrierungsergebnis bei größeren Ausreißern

- **Anpassung der Fehlermetrik** basierend auf den Punktpaaren
- **Minimierung** der Fehlermetrik

In ihrer Studie vergleichen die Autoren verschiedene Methoden anhand einer Referenzmethode [73], die bereits erfolgreich eingesetzt wurde [55]. Die Tests wurden an drei Szenarien durchgeführt: einer Wellenoberfläche (Abbildung 2.11a), einem Fraktalmuster (Abbildung 2.11b) und einer Fläche mit eingeschnittenen Kerben in Form eines „X“ (Abbildung 2.11c). Alle Szenarien wurden mit gaußschem Rauschen überlagert, um realistische Bedingungen zu simulieren.



(a) Wellenmuster (b) Fraktalmuster (c) Eingeschnittene Fläche

Abbildung 2.11: Testszenen zum Vergleich der ICP-Varianten [77]

Im Folgenden werden die verschiedenen Ansätze für jeden Schritt des Verfahrens detailliert beschrieben.

Auswahl der Punkte: Im ersten Schritt werden Punkte ausgewählt, mit denen eine Iteration durchgeführt wird.

- Die einfachste Methode besteht darin, in jeder Iteration alle verfügbaren Punkte zu verwenden. Dies führt jedoch zu einem erheblichen Rechenaufwand [12]. Um die Effizienz zu steigern, können Punkte in einem einheitlichen Raster ausgewählt werden, was den Rechenaufwand reduziert, ohne die Informationsdichte signifikant zu verringern [88].

- Ein weiterer Ansatz ist die zufällige Auswahl von Punkten, wobei die Auswahl in jeder Iteration variiert. Dies kann dazu beitragen, die Auswirkungen von Ausreißern zu minimieren [61].

- Die Autoren schlagen zudem eine Methode vor, bei der Punkte so ausgewählt werden, dass die Verteilung der Normalenvektoren möglichst breit ist. Dies ist insbesondere bei der Szene mit der eingeschnittenen Fläche vorteilhaft, da dadurch sichergestellt wird, dass in jeder Iteration Punkte aus dem Bereich der Kerben berücksichtigt werden. Bei zufälliger oder gerasterter Auswahl könnten diese kritischen Bereiche unterrepräsentiert sein.

- Zusätzlich können Punkte aus nur einer oder aus beiden Punktwolken ausgewählt werden. Um die Analogie mit den Federn fortzuführen, können aus beiden Punktwolken Punkte gewählt werden, die mit den Tangentialebenen der korrespondierenden Punkte verbunden sind [40].

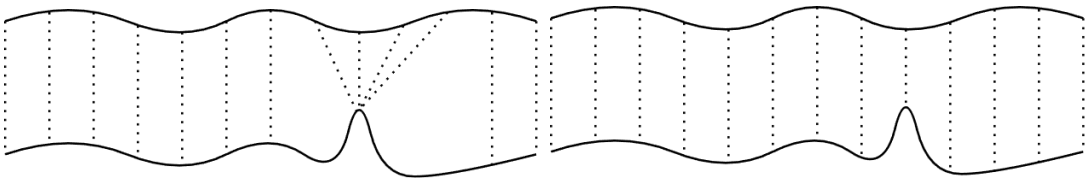
Die Ergebnisse zeigen, dass bei den Szenarien Wellen- und Fraktalmuster die verschiedenen Methoden nach wenigen Iterationen zu einer guten Registrierung konvergieren. Bei der eingeschnittenen Fläche hingegen führte nur die Methode mit gleichmäßiger Verteilung der Normalenvektoren zu zufriedenstellenden Ergebnissen. Dies liegt daran, dass Punkte außerhalb der Kerben aufgrund ihrer geometrischen Ähnlichkeit nur drei der sechs Freiheitsgrade der Transformation bestimmen können.
Ob die Punkte aus der Ausgangs-, der Zielpunktwolke oder aus beiden ausgewählt werden, hatte in diesen Experimenten keinen signifikanten Einfluss. Dies ist darauf zurückzuführen, dass die Zuordnung der Punkte auf der Basis der nächsten Nachbarn erfolgt, was eine symmetrische Behandlung beider Punktwolken gewährleistet. Bei asymmetrischen Zuordnungsverfahren könnte diese Wahl jedoch relevanter sein.

Zuordnen der Punkte: Hier werden Strategien zur Bestimmung von Korrespondenzen zwischen den Punkten der beiden Punktwolken untersucht. Die Zuordnung basiert ausschließlich auf der geometrischen Verteilung der Punkte; Farb- oder Intensitätsinformationen werden nicht berücksichtigt.

- Die einfachste und namensgebende Strategie des Algorithmus (Closest Point) ordnet jedem Punkt den nächstgelegenen Punkt in der anderen Punktwolke zu.
- Eine alternative Methode verlängert den Normalenvektor des betrachteten Punktes und bestimmt den Schnittpunkt mit der anderen Punktwolke. Der nächstgelegene Punkt zu diesem Schnittpunkt wird als korrespondierender Punkt gewählt. Dieses Verfahren wurde ebenfalls von Chen und Medioni beschrieben [17].
- Ein weiterer Ansatz nutzt die Aufnahmeposition der Kamera der Zielpunktwolke, um die Punkte der Ausgangspunktwolke auf die Zielpunktwolke zu projizieren. Ein Strahl wird vom optischen Zentrum der Kamera durch den betrachteten Punkt modelliert, und der Schnittpunkt dieses Strahls mit der Zielpunktwolke wird als Korrespondenz angenommen [14, 64].
- Eine Variante des vorherigen Verfahrens beinhaltet eine zusätzliche Suche in der Nachbarschaft des projizierten Punktes in der Zielpunktwolke nach bestimmten Merkmalen, wie minimale Punkt-zu-Punkt-Abstände oder ähnlichen Farbwerte [9].

Die Ergebnisse zeigen, dass die projektionsbasierten Verfahren im Durchschnitt bessere Ergebnisse liefern als die Methode der nächstgelegenen Nachbarn. Abbildung 2.12 verdeutlicht, dass Ausreißer bei der nächstgelegenen Nachbarschaftsmethode zu zahlreichen falschen Zuordnungen führen können, während die projektionsbasierten Verfahren robuster gegenüber Rauschen sind.
Allerdings war bei der Szene mit der eingeschnittenen Fläche die Methode der nächsten Nachbarn die einzige, die zu einer erfolgreichen Registrierung führte. Dies deutet darauf hin, dass die nächstgelegene Nachbarschaftsmethode für komplexere Szenen robuster sein kann, auch wenn sie bei einfacheren Szenarien nicht die schnellste Konvergenz bietet.

Gewichtung der korrespondierenden Punktpaare: Nachdem mit den ersten beiden Schritten die Punktpaare bestimmt wurden, können diese gewichtet werden, um



(a) Zuordnung nach nächstem Punkt (b) Zuordnung nach Projektionsverfahren

Abbildung 2.12: Verhalten der Zuordnungsverfahren bei verrauschten Daten

ihren Einfluss auf die Fehlerminimierung anzupassen. Die untersuchten Methoden umfassen:

- Gleichgewichtung aller Punktpaare ohne spezielle Anpassungen.
- Gewichtung basierend auf der Punkt-zu-Punkt-Entfernung. Punkte, die weiter als ein Schwellwert $\mathbf{Dist_{Thresh}}$ entfernt sind, erhalten ein geringeres Gewicht. Nach [40] wird das Gewicht $\mathbf{W}$ dabei folgendermaßen bestimmt:

$$\mathbf{W} = 1 - \frac{Dist(\overrightarrow{\mathbf{p_i}}, \overrightarrow{\mathbf{q_i}})}{\mathbf{Dist_{Thresh}}} \tag{2.7}$$

  Dabei stellen $\mathbf{p_i}$ und $\mathbf{q_i}$ ein Punktpaar dar. Mit der Funktion *Dist()* wird der euklidische Abstand des Punktpaares bestimmt.

- Gewichtung nach der Kompatibilität der Normalenvektoren. Das Skalarprodukt der Normalenvektoren gibt an, wie parallel die Flächen an den korrespondierenden Punkten ausgerichtet sind.
- Gewichtung basierend auf einem Unsicherheitsfaktor, der sich aus der Geometrie der Oberfläche in Bezug auf die Kameraposition ergibt. Oberflächen, die orthogonal zur Sichtlinie liegen, erhalten eine höhere Gewichtung.

Die Ergebnisse zeigten keine signifikanten Unterschiede in der Konvergenzgeschwindigkeit zwischen den verschiedenen Gewichtungsmethoden, wobei der Effekt stark vom jeweiligen Szenario abhängt.

Verwerfen von Punktpaaren: In diesem Schritt werden Punktpaare entfernt, die als unzuverlässig gelten, um den Einfluss von Ausreißern auf die Fehlerminimierung zu reduzieren. Untersuchte Methoden umfassen:

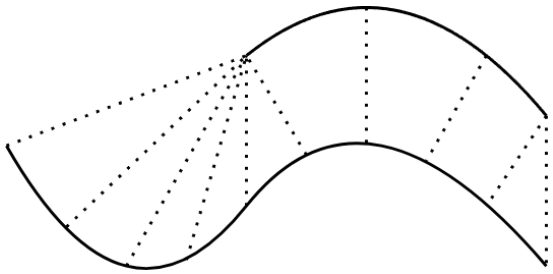


Abbildung 2.13: Fehlzuordnungen bei Grenzpunkten

- Verwerfen von Paaren, deren Abstand einen bestimmten Schwellwert überschreitet.

- Entfernen der schlechtesten 10% der Punktpaare basierend auf dem Abstand [74].

- Ausschluss von Paaren, deren Abstand mehr als das 2,5-Fache der Standardabweichung aller Abstände beträgt [61].

- Entfernen von Paaren, die inkonsistent zu benachbarten Paaren sind. Ein Paar gilt als inkonsistent, wenn gilt:

$$\mathbf{Dist_{Thresh}} \; < \; |Dist(\overrightarrow{\mathbf{p_i}}, \overrightarrow{\mathbf{p_j}}) - Dist(\overrightarrow{\mathbf{q_i}}, \overrightarrow{\mathbf{q_j}})| \tag{2.8}$$

  Wobei der Schwellenwert $\mathbf{Dist_{Thresh}}$ nach [22] definiert ist als:

$$\mathbf{Dist_{Thresh}} \; = \; 0,1 \cdot \max(Dist(\overrightarrow{\mathbf{p_i}}, \overrightarrow{\mathbf{p_j}}), \; Dist(\overrightarrow{\mathbf{q_i}}, \overrightarrow{\mathbf{q_j}})) \tag{2.9}$$

  Ein Punktepaar wird dabei jeweils von einem Punkt $\mathbf{p}$ und einem Punkt $\mathbf{q}$ gebildet.

- Ausschluss von Paaren, die Punkte an den Rändern der Punktwolken enthalten, um Fehlzuordnungen zu vermeiden [88]. Abbildung 2.13 illustriert dieses Problem.

Die meisten Methoden führten zu einer ähnlichen Konvergenzgeschwindigkeit, mit Ausnahme der Methode nach [74], die eine langsamere Konvergenz aufwies.

Fehlermetrik und Minimierung: Abschließend wurden verschiedene Fehlermetriken untersucht:

- Bei der klassischen Point-to-Point-Metrik wird die Summe der quadratischen Abstände minimiert. Für diese Metrik existieren geschlossene Lösungen, beispielsweise mittels Singulärwertzerlegung/Singular value decomposition (SVD). In [24] wurden verschiedene geschlossene Lösungen verglichen, wobei nur geringe Unterschiede in Genauigkeit und Stabilität festgestellt wurden.

- Die Point-to-Plane-Metrik summiert ebenfalls die quadratischen Fehler, jedoch existieren aufgrund der nichtlinearen Abhängigkeit von den Rotationsparametern keine geschlossenen Lösungen. Stattdessen müssen nichtlineare Optimierungsmethoden verwendet oder das Problem durch Linearisierung vereinfacht werden, beispielsweise unter der Annahme kleiner Rotationen ($\sin(\theta) \approx \theta$, $\cos(\theta) \approx 1$).

Diese Fehlermetriken können nach folgenden Strategien minimiert werden:

- Die grundlegende Methode zur Bestimmung der optimalen Transformation besteht darin, iterativ eine Transformation zu berechnen, die die Fehlermetrik minimiert, bis keine signifikante Verbesserung mehr erreicht wird.

- Eine ähnliche Vorgehensweise wird von Besl und McKay [12] beschrieben. Sie erweitern den grundlegenden ICP-Algorithmus um eine Funktionalität zur Beschleunigung der Konvergenz durch Extrapolation im Transformationsraum. Dabei analysieren sie die in den vorherigen Iterationen berechneten Transformationen, um Trends darin zu erkennen. Basierend auf diesen Trends wird die nächste Transformation extrapoliert, indem die erwartete Veränderung über das unmittelbar Berechnete hinaus projiziert wird. Durch diese Vorhersage kann der Algorithmus größere Schritte in Richtung der optimalen Ausrichtung durchführen, was die Anzahl der benötigten Iterationen zur Konvergenz signifikant reduzieren kann.

- In der Dissertation von Simon [82] wird ein Ansatz zur Verbesserung der Konvergenz des ICP-Algorithmus vorgestellt. Der Autor führt eine Methode ein, bei der in jeder Iteration mehrere zufällige Variationen der aktuellen Transformation generiert werden. Für jede dieser modifizierten Transformationen wird die Fehlermetrik berechnet, und die Variante mit dem minimalen Fehler wird für die nächste Iteration übernommen.

Die Autoren erzielen mit einem modifizierten Extrapolationsverfahren die besten Ergebnisse. Die Verwendung der Basismethode konvergiert mit der gleichen Point-to-Plane-Metrik deutlich langsamer.

Seit der ursprünglichen Einführung des ICP-Algorithmus wurden zahlreiche Verbesserungen und Anpassungen entwickelt, die die Effizienz und Genauigkeit der Registrierung von Punktwolken erhöhen. Neben den hier beschriebenen Methoden gibt es weitere Aspekte wie die Registrierung von Punktwolken aus mehreren Ansichten (Multiview-Registrierung), die spezielle Methoden erfordern. Diese werden im Verlauf der Arbeit erläutert, sofern sie relevant sind.

3 Stand der Technik

Die digitalen Fertigungsmethoden haben in den letzten Jahren bedeutende Fortschritte erzielt, insbesondere im Hinblick auf die iterative Entwicklung von Prototypen. Durch die Kombination mit Computer Aided Design (CAD)-Prozessen und den mittlerweile etablierten Methoden der additiven Fertigung, wie beispielsweise dem 3D-Druck, eröffnen sich vielfältige Möglichkeiten zur Erfassung der Objektgeometrie durch das Scannen von Bauteilen.
Dieses Kapitel bietet einen Überblick über aktuelle Lösungen zum Scannen von Objekten, die sowohl auf dem Markt verfügbar sind als auch in wissenschaftlichen Veröffentlichungen vorgestellt wurden. Dabei werden zunächst relevante Anwendungsbereiche beschrieben, gefolgt von einer Darstellung ausgewählter Lösungen im industriellen sowie im akademischen Kontext.

3.1 Anwendungsgebiete

Das Scannen von Objekten hat bereits in vielen Branchen Einzug gehalten und ist zu einem integralen Bestandteil moderner Produktions- und Entwicklungsprozesse geworden.

3.1.1 Verarbeitende Industrie

In der verarbeitenden Industrie, insbesondere in der Automobilbranche, ist die 3D-Bildgebung ein unverzichtbares Werkzeug. Der Prozess beginnt bereits beim Design der Endprodukte. So werden beispielsweise Karosserieteile teilweise manuell modelliert und anschließend mithilfe von Laserscannern digital erfasst, um sie weiterzuverarbeiten oder in CAD-Systeme zu integrieren. Darüber hinaus kommen 3D-Scanner in der Qualitätssicherung zum Einsatz, angefangen bei der Überprüfung der Abmessungen von Bauteilen bis hin zur Analyse dynamischer Verformungen während des Betriebs [81].

Ein praktisches Beispiel liefert die Volkswagen AG, die Fertigungsstraßen digitalisiert, um Modellwechsel vorzubereiten. Durch die Erfassung der bestehenden Anlagen werden potenzielle Kollisionen identifiziert und notwendige Anpassungen im Produktionsablauf vorgenommen, um einen reibungslosen Übergang zu gewährleisten [26].

3.1.2 Medizin

Auch in der Medizin gibt es zahlreiche Anwendungsfälle für die 3D-Bildgebung. Ein bemerkenswertes Beispiel ist ein Pilotprojekt in Italien, bei dem 3D-Scanner zur Herstellung von Kompressionsmasken für Patienten mit schweren Verbrennungen eingesetzt werden. Diese Masken beschleunigen die Regeneration des Hautgewebes und werden individuell auf Basis von 3D-Scans der betroffenen Region gefertigt [52].
Darüber hinaus ermöglicht die moderne Technologie die Herstellung ultrarealistischer Prothesen. So wurde beispielsweise für eine Extremsportlerin mit einem deformierten Bein eine Prothese angefertigt, indem beide Beine gescannt, das intakte Bein gespiegelt und an das deformierte Bein angepasst wurde, um ein realistisch aussehendes Gegenstück zu erzeugen [8].
Ein ähnliches Verfahren wird bei Kindern angewendet, die mit Mikrotie geboren werden, einem Zustand, bei dem ein Ohr fehlt oder missgebildet ist. Die herkömmliche Behandlungsmethode beinhaltet die Modellierung eines Ohrs aus Rippenknorpel, was mehrere Operationen erfordert und erst im Alter von etwa zehn Jahren durchgeführt werden kann. Durch den Einsatz von 3D-Scans des intakten Ohrs können jedoch realistisch aussehende synthetische Ohren hergestellt werden, was eine schonende und ästhetisch ansprechende Alternative zu invasiven chirurgischen Eingriffen bietet [62].

3.1.3 Weitere Anwendungsbereiche

Auch in anderen Bereichen wie der Kunst, der Landvermessung, der Forensik oder der Modeindustrie finden 3D-Scans vielfältige Anwendungsmöglichkeiten und haben sich als effektive Technologie etabliert. In der Kunstszene werden beispielsweise Skulpturen und Artefakte digital erfasst, um sie zu archivieren oder Repliken zu erstellen [55]. In der Forensik ermöglichen 3D-Scanner die detaillierte Dokumentation von Tatorten oder Beweismitteln. In der Modeindustrie wird die Technologie für maßgeschneiderte Bekleidung oder virtuelle Anproben genutzt.

(a) Anwendung einer Kompressionsmaske [52]

(b) Vergleich der Prothese mit dem intakten linken Bein [8]

(c) Künstliche Ohren aus dem 3D-Scann modelliert [62]

Abbildung 3.1: Beispiele von 3D-Scannern in der Medizin

3.2 Produkte und Lösungen

Im Folgenden wird ein Überblick über auf dem Markt verfügbare Lösungen gegeben. Besonders relevant sind dabei mobile Systeme, mit denen Objekte freihändig gescannt werden können. Um den Markt angemessen abzubilden, wird zunächst ein hochpreisiges professionelles Gerät und anschließend zwei vergleichsweise günstige Geräte für den Endverbraucher beschrieben. Anschließend werden Arbeiten und Veröffentlichungen beleuchtet, die sich mit dem gleichen Themengebiet befassen.

3.2.1 EinScan Pro HD

Der *EinScan Pro HD* ist ein professioneller 3D-Scanner des Herstellers Shining 3D und gehört zur Mittelklasse innerhalb des professionellen Segments. Zum Zeitpunkt der Erstellung dieser Arbeit beträgt der Preis für das System etwa 7.200 € [39]. Das Gerät verwendet die Technologie der Streifenlichtprojektion, um die Geometrie des betrachteten Objekts präzise zu erfassen. Der Scanner bietet vier verschiedene Scan-Modi, die sowohl stationäre als auch handgeführte Scans ermöglichen. Für kleine Objekte können stationäre Scans mithilfe eines Drehtellers durchgeführt werden, während für größere oder komplexere Objekte handgeführte Scans zum Einsatz kommen.
Bei handgeführten Scans können die aufgenommenen Teilbilder mithilfe von aufgeklebten Markern oder durch die Erfassung physischer Merkmale zueinander ausgerichtet werden. Laut Datenblatt erreicht der Scanner im genausten Aufnahmemodus bei Einzelaufnahmen eine Genauigkeit von bis zu 0,045 mm. Bei zusammengesetzten Aufnahmen

Abbildung 3.2: Einscan Pro HD mit aufgestecktem Color Pack [5]

beträgt die volumetrische Genauigkeit 0,3 mm/m, was bedeutet, dass die Genauigkeit des gescannten Objekts mit jedem Meter, um den das Gerät um das Objekt bewegt wird, um 0,3 mm abnimmt [6].
Mit dem optional erhältlichen *Color Pack* kann der Scanner auch die Farbtextur des Objekts erfassen und diese Informationen zur Ausrichtung der Teilbilder nutzen. Das Gerät ist mit einem LED-Projektor zur Projektion des Streifenmusters und zwei Kameras mit einer Auflösung von je 1,3 Megapixeln ausgestattet. Mit dieser Hardware können bis zu 3 Millionen Datenpunkte pro Sekunde mit einem Punktabstand von bis zu 0,2 mm aufgenommen werden. Der Scanner wiegt 1,25 kg und wird mit einer Software geliefert, die vollständige Objektscans ermöglicht und verschiedene Exportformate unterstützt [6]. Abbildung 3.2 zeigt den EinScan Pro HD mit dem Color Pack sowie das Ergebnis einer Farbaufnahme.

3.2.2 Creality CR-Scan Ferret

Der *CR-Scan Ferret* ist ein kostengünstiger 3D-Scanner, der sich an Endverbraucher richtet und vom Hersteller Creality produziert wird. Er bietet eine erschwingliche Alternative zu professionellen Geräten und ermöglicht das Scannen von kleinen bis mittelgroßen Objekten. Das Gerät nutzt die Stereokamera-Technologie und Infrarotprojektion, um die Tiefeninformationen des Objekts zu erfassen. Mit einer Tiefengenauigkeit von bis zu 0,1 mm und einer Punktauflösung von bis zu 0,16 mm bei einem Arbeitsabstand von 150 mm bis 700 mm eignet sich der Scanner für verschiedene Anwendungen, von der Modellierung kleiner Objekte bis hin zur Digitalisierung größerer Gegenstände [1].

Abbildung 3.3: CR-Scan Ferret [3]

Das 105 g schwere Gerät wird während des Scans mit einem Computer oder Smartphone verbunden, auf dem die Daten verarbeitet werden [2]. Zum Zeitpunkt der Erstellung dieser Arbeit ist die günstigste Version dieses Systems für etwa 200 € erhältlich und stellt das Einsteigermodell des Herstellers dar. In Abbildung 3.3 ist dieses Modell dargestellt.
Die zugehörige Software ermöglicht eine Echtzeitvorschau und die automatische Ausrichtung der Scans. Obwohl der CR-Scan Ferret nicht die Genauigkeit und den Funktionsumfang professioneller Geräte erreicht, bietet er Anwendern eine preisgünstige Option für den Einstieg in die 3D-Digitalisierung.

3.2.3 Creality CR-Scan Raptor

Der *CR-Scan Raptor* ist ein weiterer 3D-Scanner des Herstellers Creality, der im mittleren Preissegment angesiedelt ist. Mit einem Preis von etwa 1.600 € zum Zeitpunkt der Erstellung dieser Arbeit, bietet der *CR-Scan Raptor* eine Balance zwischen Leistung und Kosten und richtet sich sowohl an professionelle Anwender als auch an ambitionierte Hobbyisten [4].
Der Scanner verwendet die Technologie der strukturierten Lichtprojektion, um detaillierte 3D-Modelle von Objekten zu erstellen. Mit einer Einzelbildgenauigkeit von bis zu 0,05 mm und einer volumetrischen Genauigkeit von 0,1 mm/m eignet sich der Scanner

Abbildung 3.4: CR-Scan Raptor [4]

für Anwendungen, die eine hohe Präzision erfordern, wie beispielsweise Reverse Engineering, Prototyping und Qualitätskontrolle.
Die integrierte Software bietet Funktionen zur automatischen Ausrichtung und Optimierung der Scandaten, einschließlich Rauschunterdrückung, Füllung von Löchern und Texturierung. Die Ausgabeformate sind mit gängiger 3D-Software kompatibel, was den weiteren Verarbeitungsprozess erleichtert. Mit einem Gewicht von etwa 1 kg ist der CR-Scan Raptor leicht und portabel, was den Einsatz in verschiedenen Situationen ermöglicht [4]. In Abbildung 3.4 ist der CR-Scan Raptor abgebildet.

3.3 Verwandte Veröffentlichungen

Im Folgenden werden wissenschaftliche Veröffentlichungen und weitere relevante Projekte vorgestellt, die thematisch mit der vorliegenden Arbeit verwandt sind.

3.3.1 Entwicklung eines 3D-Scanners zur Qualitätssicherung

In der Veröffentlichung [92] wird die Entwicklung eines 3D-Scanners zur Qualitätssicherung in der additiven Fertigung beschrieben. Für die Anwendung ist eine sehr hohe räumliche Auflösung erforderlich. Die Arbeit konzentriert sich auf die Entwicklung und

Bewertung des bildgebenden Systems, das Punktwolken des betrachteten Objekts erzeugt.
Das System basiert auf der Verwendung von strukturiertem Licht, das von einem *AAXA P2*-Projektor auf das Objekt projiziert wird. Dieser Projektor wurde aufgrund seiner kompakten Bauform ausgewählt. Die Sensorik besteht aus zwei Kameras des Typs *FLIR GS3-U3-123S6M-C*, leistungsfähigen Industriekameras mit einer Auflösung von 4096×3000 Pixeln. Zum Zeitpunkt der Arbeit lag der Preis pro Kamera bei etwa 3.100 € [83]. Mit dem entwickelten System konnten Oberflächen mit einer Tiefenauflösung von 5 µm bei einem Sichtfeld von 15×20 mm^2 vermessen werden. Dieses Projekt stellt somit ein auf höchste Genauigkeit ausgelegtes System dar, das mit hochwertigen Komponenten in einem kleinen Bereich präzise Messergebnisse erzielt.

3.3.2 DAVID Scanner

Der *DAVID-Scanner* ist ein Projekt, das ursprünglich an der Technischen Universität Braunschweig entwickelt wurde und eine kostengünstige Lösung für 3D-Scans bietet [95]. Das System nutzt einen Linienlaser und eine Webcam, um Objekte durch Lasertriangulation zu scannen. Die Software rekonstruiert die 3D-Form des Objekts aus den aufgenommenen Bildern. Obwohl die Genauigkeit geringer ist als bei professionellen Systemen, bietet der DAVID-Scanner eine erschwingliche Möglichkeit für den Einstieg in die 3D-Digitalisierung.

3.3.3 Open-Source-Lösungen und DIY-Projekte

Neben kommerziellen Produkten gibt es auch eine Reihe von Open-Source-Projekten und DIY-Lösungen, die es ermöglichen, eigene 3D-Scanner zu bauen. Beispiele hierfür sind Projekte wie *OpenScan* [70], das auf der Verwendung von Raspberry Pi und Kameramodulen basiert, oder *FabScan* [27], ein Open-Source-Laserscanner-Projekt.
OpenScan ist ein Open-Source-Projekt von Thomas Megel. Mit 3D-gedruckten Komponenten, einem Raspberry Pi und der frei verfügbaren Software können kleine Objekte mit einer Genauigkeit von bis zu 0,01 mm gescannt werden. Auf der Projektwebsite können alle benötigten Teile zum Bau eines solchen Scanners erworben oder die notwendigen Anleitungen und Fertigungsdaten heruntergeladen werden.
FabScan basiert auf der Abschlussarbeit von Francis Engelmann [28] und wurde in verschiedenen Iterationen kontinuierlich weiterentwickelt. Die zum Zeitpunkt dieser Arbeit

aktuellste Version geht auf die Abschlussarbeit von Mario Lukas zurück [59]. Auf der Projektseite stehen Bauanleitungen, Software und Materiallisten zur Verfügung, die den Aufbau eines eigenen 3D-Scanners ermöglichen.

4 Anforderungsanalyse

In diesem Kapitel werden die Anforderungen definiert, die für eine erfolgreiche Entwicklung des geplanten 3D-Scanners erfüllt werden müssen. Zunächst werden die Stakeholder identifiziert, also die Personen und Personengruppen, die in irgendeiner Weise mit dem System interagieren oder von ihm betroffen sind. Auf Grundlage dieser Analyse werden anschließend die Anwendungsfälle beschrieben, die die Funktionalitäten des Systems aus Sicht der Benutzer darstellen.
Basierend auf diesen Erkenntnissen wird im weiteren Verlauf das System spezifiziert. Dabei werden die verschiedenen Teilsysteme detailliert beschrieben und ihre Funktionen abgegrenzt, um festzustellen, welche Komponenten erforderlich sind. Abschließend werden klare und messbare Anforderungen formuliert, die sowohl als Grundlage für die folgende Entwicklungsphase dienen als auch die Bewertungskriterien festlegen, anhand derer das System letztendlich beurteilt wird.

4.1 Stakeholder

Eine erfolgreiche Systementwicklung setzt voraus, dass die Interessen und Bedürfnisse aller beteiligten Personengruppen umfassend berücksichtigt werden. Im Folgenden werden daher die relevanten Stakeholder identifiziert und ihre Interessen an dem zu entwickelnden System beschrieben.

Auftraggeber

Der Auftraggeber, Prof. Dr. Hensel, in seiner Funktion als Erstprüfer dieser Arbeit, strebt eine Erweiterung des Wissens im betreffenden Themengebiet an. Ein besonderes Interesse liegt dabei in der Verarbeitung von freihändig erfassten 3D-Daten, insbesondere der Punktwolkenverarbeitung. Konkret umfasst dies die Prozesskette von der Aufnahme der Daten, der Datenvorverarbeitung, die Registrierung der Punktwolken bis

zur Erzeugung einer geschlossenen Oberfläche und der daraus folgenden Erstellung eines vollständigen 3D-Modells.
Das übergeordnete Ziel besteht darin, Modelle realer Objekte zu erstellen, die mithilfe eines 3D-Druckers reproduziert werden können. Darüber hinaus soll das zu entwickelnde System als Demonstrator im Hochschulkontext dienen und für Lehrzwecke in Vorlesungen genutzt werden können.

Entwickler

Der Entwickler des Systems ist zugleich Verfasser dieser Arbeit. Sein primäres Ziel ist ein erfolgreicher Abschluss der Masterarbeit, was bedeutet, dass die übergeordneten Ziele und die im Folgenden definierten Anforderungen möglichst umfassend erfüllt werden. Darüber hinaus strebt er an, sein fachliches Wissen und seine Fähigkeiten zu erweitern sowie seine Kompetenzen im Projektmanagement zu vertiefen.

Nutzer erster und zweiter Instanz

Die Anwender des Systems lassen sich in Nutzer erster und zweiter Instanz unterteilen, basierend auf ihren Anforderungen und den erzeugten Ausgabedaten.
Nutzer erster Instanz erwarten, dass das System ein präzises dreidimensionales Abbild eines realen Objekts generiert, welches in einem CAD-Programm weiterbearbeitet oder mithilfe eines 3D-Druckers physisch reproduziert werden kann. Ihre primäre Anforderung besteht darin, dass das erstellte Abbild möglichst genau den Dimensionen des realen Objekts entspricht und alle Details exakt darstellt.
Nutzer zweiter Instanz verwenden das generierte Modell für Funktionsprüfungen von Bedienfeldern in Luftfahrzeugen. Während des Scanvorgangs wird parallel die Oberflächentextur des realen Objekts erfasst, welche am Ende des Prozesses auf das virtuelle Modell übertragen wird. Mit diesem texturierten 3D-Modell sollen automatisierte Gerätetests durchgeführt werden, bei denen Benutzerschnittstellen wie Schalter und Anzeigen auf ihre Funktionalität geprüft werden. Das 3D-Modell dient dazu, den Typ und die exakte Position der zu testenden Schalter zu bestimmen. Die Texturdaten dienen zur genaueren Identifikation des Schnittstellentyps und der Definition von Schalterpositionen durch die Analyse von Beschriftungen. Auf Basis der Daten soll im Nachgang ein Ablaufprogramm für einen Roboterarm erstellt werden, der die Positionen anfahren und die Betätigung testen kann. Diese Anwendergruppe legt besonderen Wert auf

exakte Objektdimensionen und die präzise Positionierung der Textur. Alle Anwender erwarten darüber hinaus intuitiv bedienbare Nutzerschnittstellen, die einen effizienten Arbeitsfluss ermöglichen.

Folgeentwickler

Es ist wahrscheinlich, dass die Ergebnisse dieser Arbeit sowohl im akademischen als auch im industriellen Kontext von Folgeentwicklern genutzt werden. Im akademischen Bereich könnten weitere Abschlussarbeiten im Bereich der 3D-Bildgebung entstehen, die auf den Ergebnissen oder Ansätzen dieser Arbeit aufbauen. Insbesondere ist eine Weiterentwicklung im Kontext der Teststandentwicklung für Luftfahrzeugkomponenten denkbar. Beide Gruppen, sowohl im akademischen als auch im industriellen Umfeld, legen Wert auf eine übersichtliche und vollständige Dokumentation. Darüber hinaus erwarten sie gut kommentierten und strukturierten Quellcode, in dem alle Schnittstellen und Übergabewerte ausführlich beschrieben sind.

Betreiber des Systems

Es besteht die Möglichkeit, dass das System potenziell gefährliche Komponenten wie Laser enthält, die für den Anwender gefährlich sein können. Daher ist es von höchster Bedeutung, jegliche Risiken für Personenschäden auszuschließen. Betreiber des Systems haben ein großes Interesse daran, dass im Rahmen der Entwicklung alle geltenden Normen zur Unfallverhütung eingehalten und überprüft werden, um die Sicherheit zu gewährleisten.

4.2 Anwendungsfälle

Basierend auf der zuvor durchgeführten Stakeholderanalyse werden in diesem Abschnitt die relevanten Anwendungsfälle des Systems analysiert. Abbildung 4.1 zeigt das zugehörige Anwendungsfalldiagramm, das die wesentlichen Anwendungsfälle darstellt, die zur Erreichung der Ziele erforderlich sind. Nach einer kurzen Erläuterung des Diagramms werden alle Anwendungsfälle beschrieben, um einen Überblick über die Funktionalitäten des Systems zu gewährleisten.

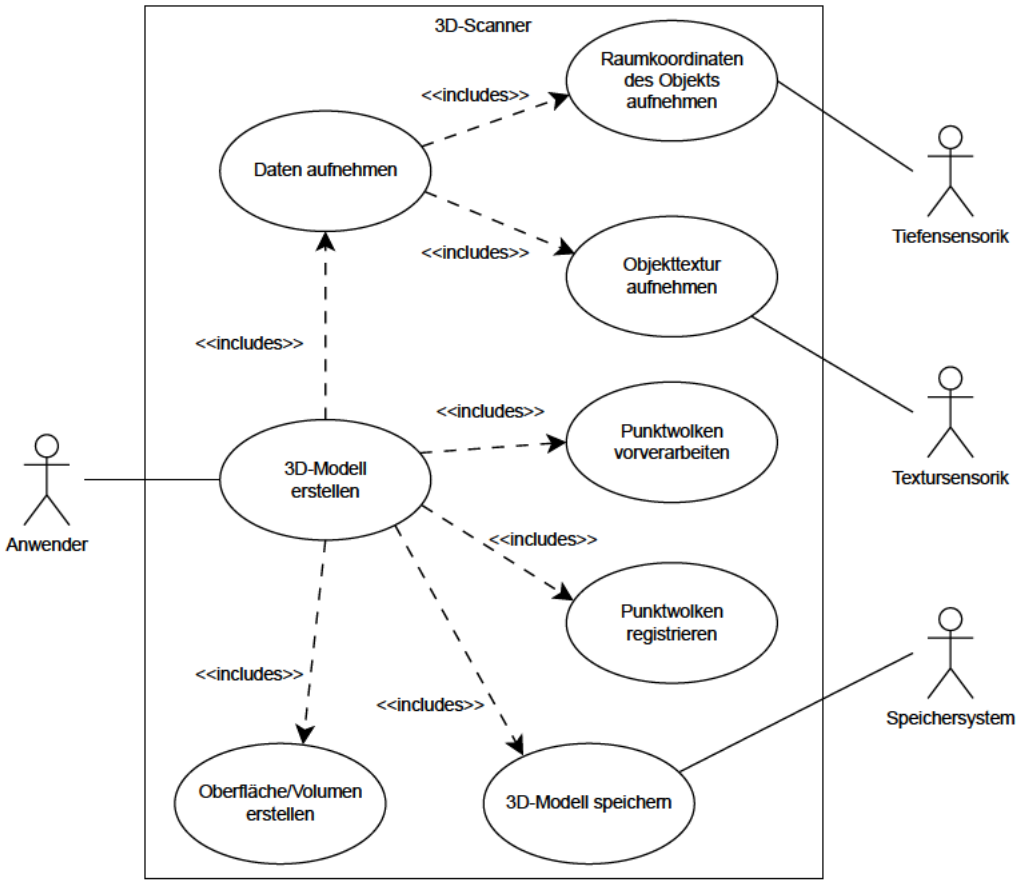


Abbildung 4.1: Anwendungsfalldiagramm des 3D-Scanners

Das System „3D-Scanner“ betrachtet den Benutzer als Akteur, der die dargestellten Anwendungsfälle im Kontext eines definierten Ablaufs initiiert. Der Benutzer ist zudem an der Änderung der Perspektive beteiligt, da der Scanner manuell geführt wird. Weiterhin agieren die Sensorik, die bei der Datenaufnahme involviert ist, sowie das Speichersystem, das bei Lade- und Speicheroperationen mit dem System interagiert, als Akteure. Der übergeordnete Anwendungsfall ist „3D-Modell erstellen“, der alle weitere Anwendungsfälle inkludiert, die während der Ausführung ebenfalls durchlaufen werden müssen. Der Anwendungsfall „Daten aufnehmen“ inkludiert zudem weitere spezifische Anwendungsfälle, die zur vollständigen Datenerfassung notwendig sind. Im Folgenden werden alle Anwendungsfälle detailliert beschrieben. Für jeden Anwendungsfall werden die Vorbedingungen und die Nachbedingungen definiert.

Anwendungsfall 1: 3D-Modell erstellen

Dieser Anwendungsfall umfasst den gesamten Prozess der 3D-Modellerstellung, einschließlich der Datenerfassung, Datenverarbeitung und Modellgenerierung. Der Benutzer führt alle notwendigen Schritte durch, um ein digitales Abbild des realen Objekts zu erhalten.
Vorbedingungen: Das System ist betriebsbereit, und es läuft aktuell kein anderer Modellierungsprozess.
Nachbedingungen: Ein vollständiges 3D-Modell des gewünschten Objekts wurde erstellt und gespeichert.

Daten aufnehmen

Dieser Anwendungsfall umfasst alle Aktivitäten, die zur Erfassung der notwendigen Daten führen. Er inkludiert die Anwendungsfälle „Raumkoordinaten des Objekts aufnehmen“, „Objekttextur aufnehmen“ und „Perspektive auf das Objekt verändern“.
Vorbedingungen: Aktuell sind keine Daten im Programmspeicher vorhanden, oder wurden bereits verarbeitet.
Nachbedingungen: Punktwolken und Texturdaten des Objekts aus allen erforderlichen Perspektiven liegen im Programmspeicher vor.

Anwendungsfall 2.1: Raumkoordinaten des Objekts aufnehmen

Die Sensorik des Systems erfasst die Oberflächengeometrie des Objekts aus der aktuellen Perspektive, wobei eine Punktwolke generiert wird, die die räumliche Anordnung der erfassten Punkte beschreibt.
Vorbedingungen: Aus der aktuellen Position wurden noch keine Raumkoordinaten des Objekts aufgenommen.
Nachbedingungen: Eine Punktwolke des Objekts wurde aus der aktuellen Position aufgenommen und im Programmspeicher abgelegt.

Anwendungsfall 2.2: Objekttextur aufnehmen

Die Farbinformationen der Objektoberfläche werden aus der aktuellen Perspektive erfasst.

Vorbedingungen: Aus der aktuellen Position wurde noch keine Textur des Objekts aufgenommen.
Nachbedingungen: Die Texturdaten der aktuellen Position wurden aufgenommen und im Programmspeicher gespeichert.

Anwendungsfall 3: Punktwolken vorverarbeiten

Dieser Anwendungsfall umfasst die Anwendung von Filter- und Optimierungsalgorithmen auf die aufgenommenen Punktwolken und Texturdaten, um die Datenqualität zu erhöhen und die Verarbeitungszeit zu reduzieren.
Vorbedingungen: Punktwolken und Texturdaten wurden aufgenommen und liegen im Programmspeicher vor.
Nachbedingungen: Die Daten wurden vorverarbeitet und für die Registrierung optimiert.

Anwendungsfall 4: Punktwolken registrieren

Mittels geeigneter Algorithmen, wie beispielsweise dem Iterative Closest Point (ICP)-Algorithmus, werden die Punktwolken in ein gemeinsames Koordinatensystem überführt, sodass sie ein kohärentes Modell des Objekts bilden.
Vorbedingungen: Punktwolken und Texturdaten wurden erzeugt und vorverarbeitet.
Nachbedingungen: Die Einzelpunktwolken wurden erfolgreich zu einer Gesamtpunktwolke zusammengefügt.

Anwendungsfall 5: Oberfläche und Volumen erstellen

Durch Anwendung von Meshing-Algorithmen, wie zum Beispiel mit der *Poisson Surface Reconstruction* [51] oder der *Delaunay-Triangulation* [54], wird aus der Punktwolke eine polygonale Oberfläche erzeugt, die das Objekt repräsentiert. Aus dieser wird wiederum ein Volumenmodell abgeleitet, das für Simulationen oder Fertigungsprozesse genutzt werden kann.
Vorbedingungen: Die registrierte Punktwolke liegt im Programmspeicher vor.
Nachbedingungen: Eine zusammenhängende Oberfläche und das daraus resultierende Volumenmodell wurden erzeugt.

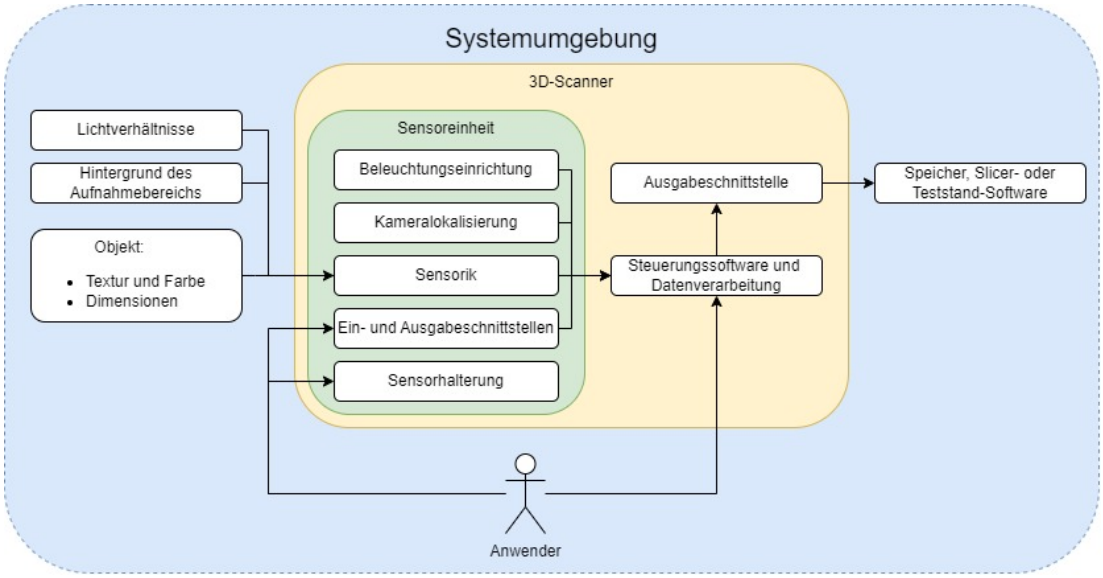


Abbildung 4.2: Systemumgebung des 3D-Scanners

Anwendungsfall 6: 3D-Modell speichern

Das Modell wird in einem gängigen Dateiformat (etwa `.stl` oder `.obj`) gespeichert, um es in anderen Anwendungen zu nutzen oder zu archivieren.
Vorbedingungen: Ein 3D-Modell wurde erzeugt.
Nachbedingungen: Das 3D-Modell wurde auf dem Computer im gewünschten Format gespeichert.

4.3 Systemdefinition

Basierend auf den Anforderungen der Stakeholder und den definierten Anwendungsfällen wird in diesem Abschnitt der Aufbau des Systems detailliert beschrieben. Hierzu werden die Komponenten identifiziert und erläutert, die für die korrekte Funktion des Systems erforderlich sind. Abbildung 4.2 visualisiert das System mit seinen Teilkomponenten innerhalb seines Umfelds sowie mögliche Schnittstellen und Einflüsse. Im Folgenden werden die in der Abbildung dargestellten Komponenten beschrieben.

4.3.1 Sensoreinheit

Die erste Hauptkomponente des Systems ist die **Sensoreinheit**, die aus mehreren Unterkomponenten besteht und den Kontakt zur Umwelt herstellt. Zentrales Element der Sensoreinheit ist die **Sensorik**, welche für die Erfassung der notwendigen Daten verantwortlich ist. Um ein präzises 3D-Modell erstellen zu können, ist eine leistungsfähige Datenerfassung erforderlich, die sowohl die geometrischen Dimensionen des Objekts als auch dessen Textur erfasst.
Ein entscheidender Faktor, der die Qualität der Texturaufnahme beeinflusst, sind die äußeren **Lichtverhältnisse**. Bei unzureichender Beleuchtung kann die Textur möglicherweise gar nicht aufgenommen werden, während eine ungleichmäßige Ausleuchtung zu variierenden Farbdarstellungen aus unterschiedlichen Perspektiven führt. Dies kann letztlich zu einer ungleichmäßigen Texturierung des 3D-Modells führen. Um diesem Problem entgegenzuwirken, ist die Integration einer **Beleuchtungseinrichtung** sinnvoll, die das Objekt aus jeder Position gleichmäßig ausleuchtet und somit optimale Bedingungen für die Sensorik schafft.
Die Qualität der gesamten Datenerfassung wird zudem maßgeblich von den **physikalischen Eigenschaften** der zu scannenden Objekte beeinflusst, insbesondere von Farbe, Material und Oberflächenbeschaffenheit. Diese Eigenschaften haben direkten Einfluss auf die Auswahl der geeigneten Sensortechnologie. Optische Messverfahren basieren auf der Reflexion von Licht an der Objektoberfläche in Richtung des Sensors. Die spezifischen Objekteigenschaften bestimmen dabei, wie Licht unterschiedlicher Wellenlängen reflektiert wird. Da verschiedene Sensortechnologien mit unterschiedlichen Lichtquellen arbeiten, beeinflusst dies die Fähigkeit der Technologie, ein Objekt adäquat zu erfassen. Für die Anwender der ersten Instanz ist ein möglichst breiter Anwendungsbereich wünschenswert, da keine spezifischen Materialeigenschaften der zu scannenden Objekte vorausgesetzt werden können. Im Gegensatz dazu arbeiten die Anwender der zweiten Instanz mit Objekten, die stets ähnliche Oberflächeneigenschaften aufweisen. Abbildung 4.3 zeigt ein Beispiel eines zu scannenden Panels. Die Oberfläche ist mit einem grauen oder blauen, matten Lack beschichtet, und Schalter sowie Anzeigen bestehen überwiegend aus Kunststoff. Eine Herausforderung stellen kleine Metalloberflächen dar, wie sie bei Kippschaltern vorkommen, da diese projizierte Muster oder Laser reflektieren und somit schwer zu erfassen sind.
Der **Aufnahmebereich**, also der Bereich um das Objekt herum, beeinflusst ebenfalls die Qualität der aufgenommenen Daten. Es ist wichtig, dass sich das Objekt ausreichend vom Hintergrund abhebt, um eine eindeutige Differenzierung zwischen Objekt und Um-

gebung zu ermöglichen. Dies kann durch den Einsatz kontrastreicher Hintergründe oder spezieller Materialien erreicht werden.
Da das System freihändig aufgenommene Daten verwendet, ist eine Möglichkeit zur **Kameralokalisierung** während der Aufnahme von Vorteil, um die Genauigkeit des Systems und die Effizienz der Datenverarbeitung zu verbessern. Insbesondere bei der Verarbeitung von 3D-Daten entstehen aufgrund der hohen Datenmenge erhebliche Rechenzeiten. Um die erforderliche Rechenzeit zur Registrierung der Punktwolken zu minimieren, sollten diese im Voraus möglichst präzise zueinander positioniert werden. Ein bewährter Ansatz zur Erfassung der Kameraposition ist die Verwendung einer Inertial Measurement Unit (IMU). Diese misst Beschleunigungen und Drehraten, wodurch die Bewegung der Kamera abgeschätzt werden kann. Eine weitere Möglichkeit besteht darin, die generierten Bild- und Tiefendaten zur visuellen Odometrie zu nutzen. Die relative Kameraposition kann aus aufeinanderfolgenden Bildern oder durch den Einsatz von Markern bestimmt werden.
Die Sensoreinheit stellt zudem eine Benutzerschnittstelle bereit. Zum einen wird sie in einer ergonomisch gestalteten **Sensorhalterung** integriert, die vom Anwender bewegt und positioniert wird. Da die Datenaufnahme manuell erfolgt, darf die Sensoreinheit nicht zu schwer sein, um eine Ermüdung des Benutzers während des Scanvorgangs zu vermeiden. Es ist daher notwendig, einen Kompromiss zwischen Funktionalität und Gewicht zu finden, um eine ergonomische Bedienung zu gewährleisten.
Zum anderen kann die Sensoreinheit mit **Bedienelementen** ausgestattet werden, die eine direkte Interaktion mit dem System ermöglichen. Dies könnten beispielsweise Tasten oder Schalter sein, mit denen der Benutzer den Scanvorgang steuert, ohne die Handhabung der Sensoreinheit zu unterbrechen. Diese direkte Steuerungsmöglichkeit fördert eine intuitive Bedienung.

4.3.2 Steuerungssoftware und Datenverarbeitung

Die zweite Hauptkomponente des Systems ist die **Steuerungssoftware und Datenverarbeitung**, die mit der Sensoreinheit kommuniziert. Die Hauptaufgabe dieser Software besteht darin, die aufgenommenen Teilbilder des Objekts zu verarbeiten und mittels Verfahren zur Punktwolkenregistrierung zu einem konsistenten Gesamtbild zusammenzufügen. Um ein optimales Ergebnis zu erzielen, ist es oft notwendig, die Teilbilder vorab zu optimieren und zu filtern, um Rauschen zu reduzieren und nicht relevante Informationen zu entfernen. Aus der zusammengesetzten Punktwolke wird anschließend

Abbildung 4.3: Nachbau eines A320-Autopilotpanels [25]

mittels Meshing-Algorithmen eine Oberfläche generiert. Verfahren wie die Poisson Surface Reconstruction [51] oder die Delaunay-Triangulation [54] können hier Anwendung finden, um eine geschlossene Oberfläche zu erzeugen. Abschließend wird aus der Oberfläche ein Volumenmodell erzeugt und gespeichert.
Die Steuerungssoftware implementiert zudem die Benutzerschnittstelle in Form einer Computeranwendung mit grafischer Oberfläche. Alle in Kapitel 4.2 definierten Anwendungsfälle können vom Anwender über diese Schnittstelle initiiert werden, wodurch der Programmablauf gesteuert wird. Die Benutzeroberfläche sollte dabei intuitiv gestaltet sein und dem Anwender Feedback zum aktuellen Prozessstatus geben.
Darüber hinaus stellt die Software Schnittstellen zu weiteren Softwarediensten bereit. So kann das erzeugte 3D-Modell in verschiedenen Datenformaten exportiert werden, um es beispielsweise für den 3D-Druck oder in CAD-Programmen weiterzuverarbeiten. Wenn die Textur des Objekts für die Anwendung relevant ist, kann das Modell in einem Format gespeichert werden, das Texturinformationen unterstützt, wie zum Beispiel `.obj` oder `.ply`.

4.4 Anforderungen

Nachdem die relevanten Stakeholder identifiziert, die Anwendungsfälle analysiert und die Systemkomponenten definiert wurden, können daraus die spezifischen Anforderungen an das System abgeleitet werden. Diese Anforderungen werden in funktionale Anforderungen (F) und nicht-funktionale Anforderungen (NF) unterteilt. Funktionale Anforderungen beschreiben die vom System zu erfüllenden Aufgaben, während nicht-funktionale Anforderungen Qualitätskriterien oder Bedingungen festlegen, unter denen die funktionalen Anforderungen erfüllt werden müssen.

4.4.1 Hardwareanforderungen

Die Hardwareanforderungen werden im Folgenden erläutert und in generelle Anforderungen (GE), Anforderungen an die Sensoreinheit (SE) und Anforderungen an den Aufnahmebereich (AB) unterteilt.

Generelle Anforderungen

HW-GE 1 (NF): Die Kosten für die Neuanschaffung von Hardwarekomponenten dürfen 250 € nicht überschreiten.
Dieser Wert ergibt sich aus den Richtlinien des Departments für Informations- und Elektrotechnik an der Hochschule für Angewandte Wissenschaften (HAW) Hamburg für interne Abschlussarbeiten.

HW-GE 2 (NF): Das System muss während des Betriebs sicher sein und darf keine Gefahr für Personen darstellen. Potenzielle Gefahren sind zu identifizieren und gemäß geltender Normen und Vorschriften zu minimieren.

4.4.2 Anforderungen Sensoreinheit

HW-SE 1 (F): Die Sensoreinheit erfasst die Oberflächengeometrie des Objekts.

HW-SE 2 (F): Die Sensoreinheit erfasst die Textur des Objekts.

HW-SE 3 (F): Die Sensoreinheit ermöglicht die freihändige Bewegung um das gesamte Objekt.

HW-SE 4 (NF): Die Sensoreinheit wiegt einschließlich aller Komponenten weniger als 400 Gramm.
Gemäß [56] sollte das Gewicht von Werkzeugen für Präzisionsarbeiten nicht mehr als 400 g betragen, um Ermüdungserscheinungen beim Benutzer zu vermeiden.

HW-SE 5 (NF): Das System ist in der Lage, Objekte mit matten Oberflächen aus Metall oder Kunststoff zu erfassen, die über Schaltelemente und Anzeigen verfügen.
Bei den automatisierten Gerätetests werden überwiegend Flugzeugbedienfelder gescannt, deren Oberflächen matt lackiert sind. Die darauf befindlichen Bedienelemente bestehen in der Regel aus Kunststoff oder, im Fall von Kippschaltern, aus reflektierendem Metall.

HW-SE 6 (F, Optional): Der Zustand des Scanvorgangs kann über Bedienelemente an der Sensoreinheit gesteuert werden.
Der Benutzer kann den Aufnahmeprozess durch Taster an der Sensoreinheit starten und beenden. Außerdem kann er die Bildaufnahme direkt steuern, indem er wie bei einer Kamera einen Auslöser betätigt. Gegebenenfalls können Anzeigen dem Benutzer Feedback geben.

HW-SE 7 (F, Optional): Die Sensoreinheit verfügt über eine integrierte Beleuchtungseinrichtung.

4.4.3 Anforderungen Aufnahmebereich

HW-AB 1 (NF): Das System unterstützt einen Aufnahmebereich um Objekte mit den maximalen Abmessungen von 60 cm × 60 cm × 25 cm aufzunehmen.
Als größtes zu scannendes Objekt wird das in Abbildung 4.3 gezeigte Autopilotpanel festgelegt, das in beliebiger Ausrichtung aufgenommen werden kann. Seine größte Dimension beträgt 59 cm in der Breite. Die maximale Höhe wurde zu 25 cm festgelegt. Damit sind die Dimensionen der voraussichtlichen Objekte mit einem zusätzlichen Puffer abgedeckt.

4.4.4 Softwareanforderungen

Die Softwareanforderungen werden im Folgenden aufgeführt und in generelle Anforderungen (GE), Anforderungen an die Datenaufnahme (DA), Anforderungen an die Datenverarbeitung (DV) sowie Anforderungen an die Bedienoberfläche (BO) unterteilt.

4.4.5 Generelle Anforderungen

SW-GE 1 (NF): Jedes Modul der Software muss über eine aussagekräftige Dokumentation verfügen.
Das genaue Format wird abhängig von der Programmiersprache mit dem Betreuer dieser Arbeit abgestimmt, um ein einheitliches Format zwischen dieser und anderen Abschlussarbeiten zu gewährleisten.

SW-GE 2 (NF): Das System erzeugt digitale Abbilder von realen Objekten mit einer maximalen Abweichung von ±0,2 mm zum realen Objekt.
Basierend auf Untersuchungen des Herstellers beträgt die Abweichung eines 3D-gedruckten Objekts auf einem Ultimaker S5 maximal ±0,2 mm [90]. Da von den Benutzergruppen keine konkreten Anforderungen an die Genauigkeit vorgegeben sind, wird dieser Wert als Zielvorgabe festgelegt.

SW-GE 3 (F): Die Software ist auf einem Computer mit dem Betriebssystem Windows 10 lauffähig.
Zum Zeitpunkt der Erstellung dieser Arbeit lag der Marktanteil von Windows bei den führenden Betriebssystemen bei etwa 72,5 %, was eine Entwicklung für Windows-Computer nahelegt [84].

SW-GE 4 (NF): Die Datenverarbeitung vom Abschluss der Aufnahme bis zur Ausgabe des fertigen Modells soll auf einem Computer mit einem Windows-Leistungsindex von mindestens sieben innerhalb von maximal $2+\frac{n}{2}$ Minuten abgeschlossen sein, wobei n der Anzahl der aufgenommenen Perspektiven entspricht.
Zur Festlegung einer maximalen Verarbeitungsdauer dient die Anzahl der aufgenommenen Daten als Grundlage. Jeder Verarbeitungsschritt muss auf jede erfasste Punktwolke angewandt werden, wodurch die Gesamtverarbeitungszeit proportional zur Anzahl der aufgenommenen Perspektiven ansteigt. Um die Wartezeit für den Benutzer zu minimieren, ist eine Grundverarbeitungszeit von zwei Minuten vorgesehen, die potenzielle Verzögerungen durch Benutzereingaben abdeckt. Für jede zusätzliche Perspektive werden weitere 30 Sekunden eingeplant. Der angegebene Windows-Leistungsindex basiert auf dem in dieser Arbeit eingesetzten Computer und kann auf jedem Computer mit einem Windows-10-Betriebssystem ermittelt werden.

4.4.6 Anforderungen an die Datenaufnahme

SW-DA 1 (F): Die Änderung der Kameraposition zwischen den aufgenommenen Perspektiven kann nach der Datenaufnahme ermittelt werden.
Zur globalen Vorregistrierung der Punktwolken wird die aus den aufgenommenen Daten ermittelte Kameraposition verwendet.

SW-DA 2 (F, Optional): Die Beleuchtungseinrichtung kann während der Datenaufnahme vom steuernden Computer in ihrer Intensität angepasst werden, um das Objekt aus allen Richtungen gleichmäßig auszuleuchten.

SW-DA 3 (F, Optional): Während der Datenaufnahme erhält der Benutzer Feedback zum aktuellen Abstand der Sensoreinheit zum Objekt.
Optische Datenerfassungssysteme haben häufig einen optimalen Arbeitsabstand bzw. Abstände, in denen keine zuverlässigen Daten erhoben werden können. Ein Feedbacksystem soll den Anwender unterstützen, den optimalen Bereich nicht zu verlassen.

SW-DA 4 (F, Optional): Während der Datenaufnahme erhält der Benutzer Feedback zum aktuellen Fortschritt des Scanvorgangs.
Um sicherzustellen, dass das Objekt von allen Seiten in ausreichender Qualität aufgenommen wird, kann ein Feedbacksystem zum Fortschritt des Scans implementiert werden.

4.4.7 Anforderungen an die Datenverarbeitung

SW-DV 1 (F): Die Software speichert Objekte zur Reproduktion mittels 3D-Drucker im `.stl`-Format.

SW-DV 2 (F): Die Software speichert Objekte zur digitalen Weiterverarbeitung im *.obj*-Format.

4.4.8 Anforderungen an die Bedienoberfläche

SW-BO 1 (F): Der Prozess der 3D-Modellerstellung wird durch eine grafische Benutzeroberfläche gesteuert.

SW-BO 2 (F): Um ein Bewusstsein für den aktuellen Programmstand zu schaffen, werden in jedem Schritt des Programms die aktuellen Daten in einer 2D- oder 3D-Anzeige dargestellt.

SW-BO 3 (NF): Die angezeigten dreidimensionalen Daten können rotiert und skaliert werden.
Erste Untersuchungen mit Punktwolken im Rahmen dieser Arbeit haben gezeigt, dass statische 3D-Darstellungen auf dem Bildschirm schwer zu interpretieren sind, wenn keine Interaktionen wie Rotation oder Skalierung möglich sind.

5 Konzept

In diesem Kapitel wird ein detailliertes Konzept entwickelt, das den in Kapitel 4.4 festgelegten Anforderungen gerecht wird. Dabei wird die Vorgehensweise für die nachfolgende Entwicklung erläutert. Obwohl der Schwerpunkt dieser Arbeit auf der Softwareentwicklung liegt, wird zunächst die Auswahl und Beschreibung der zugrunde liegenden Hardware durchgeführt um die Rahmenbedingungen für das darauf aufbauende Softwarekonzept zu definieren. Anschließend werden entscheidende Entwicklungsentscheidungen dargestellt und begründet. Diese Entscheidungen bilden die Grundlage für die Implementierung. Nachdem alle grundlegenden Entscheidungen getroffen wurden, wird das Konzept für die Softwareentwicklung beschrieben. Dieses Konzept umfasst die Architektur der Software, die Auswahl geeigneter Algorithmen sowie die geplanten Tests und Validierungen, um sicherzustellen, dass das entwickelte System den gesetzten Zielen und Anforderungen entspricht.

5.1 Sensorik

In diesem Abschnitt wird ein geeigneter Sensor zur Erfassung der Objektdaten ausgewählt. Zunächst werden die möglichen Technologien zur Erfassung der 3D-Daten analysiert und bewertet. Anschließend wird unter Berücksichtigung der verfügbaren Produkte eine konkrete Auswahl getroffen, die den in Abschnitt 4.4 definierten Anforderungen entspricht.

5.1.1 Vergleich optischer 3D-Datenerfassungsmethoden

Es existiert eine Vielzahl von Verfahren zur optischen 3D-Datenerfassung (siehe Abbildung 2.1 in Kapitel 2.1). Für die Entwicklung eines handgeführten 3D-Scanners, der den Anforderungen entspricht, werden im Folgenden zwei der am häufigsten verwendeten Ansätze analysiert und verglichen: die Stereophotogrammetrie und die aktiven

Projektionsverfahren. Beide Technologien sind bei entsprechender Implementierung in der Lage, eine hohe Genauigkeit zu liefern und sind in handelsüblichen 3D-Scannern weit verbreitet (siehe Kapitel 3.2).
Die Entscheidung zwischen diesen beiden Methoden wird durch eine Reihe von Faktoren beeinflusst, die sich aus den spezifischen Anforderungen der Anwendung ableiten. Neben der Erfassungsgenauigkeit sind Aspekte wie die Handhabbarkeit während des Betriebs, die Ergonomie und die Fähigkeit zur Integration in ein mobiles, handgeführtes System gemäß **HW-SE 3**, **HW-SE 4** und **HW-SE 6** entscheidend.
Zunächst werden die beiden Methodiken nach [13, 37] beschrieben. Die in Tabelle 5.1 dargestellten Bewertungskriterien dienen anschließend zur Beurteilung der Eignung der beiden Verfahren für die vorliegende Arbeit. Das Ziel dieses Vergleichs ist es, die Methode zu identifizieren, die den Anforderungen des Projekts am besten gerecht wird. Die Bewertung erfolgt unter Berücksichtigung der Rahmenbedingungen dieser Arbeit, insbesondere des verfügbaren Budgets und der begrenzten industriellen Kontakte. Dies betrifft vor allem die Beschaffung von Komponenten wie miniaturisierten Projektoren und hochauflösenden Industriekameras, falls ein eigenes Sensorsystem aufgebaut werden soll.

Stereophotogrammetrie

Die Stereophotogrammetrie ist ein Verfahren, bei dem zwei oder mehr Kameras verwendet werden, um ein Objekt aus unterschiedlichen Blickwinkeln zu erfassen. Durch die Auswertung der Disparität zwischen den aufgenommenen Bildern können präzise 3D-Daten generiert werden [13]. Im Gegensatz zu anderen 3D-Scanning-Methoden verzichtet die Stereophotogrammetrie auf die aktive Projektion von Lichtmustern und nutzt stattdessen das vorhandene Umgebungslicht.
Ein entscheidender Vorteil dieser Methode liegt in ihrem geringen Formfaktor, der eine hohe Mobilität des Systems ermöglicht. Die gesamte Apparatur kann kompakt und leicht gestaltet werden, was die Handhabung, insbesondere bei handgeführten Anwendungen, deutlich vereinfacht (**HW-SE 4**). Darüber hinaus bietet die Stereophotogrammetrie ein hohes Maß an Sicherheit, da keine Laser oder andere potenziell gefährlichen Lichtquellen zum Einsatz kommen (**HW-GE 2**), wodurch keine Gefahr für den Benutzer oder umstehende Personen besteht.
Insgesamt zeichnet sich die Stereophotogrammetrie durch ihre Mobilität, Benutzer-

freundlichkeit und Sicherheit aus, was sie zu einer attraktiven Wahl für die vorliegende Anwendung im Bereich der 3D-Datenerfassung macht.

Aktive Projektionsverfahren

Aktive Projektionsverfahren nutzen Projektoren, um spezifische Lichtmuster auf das zu scannende Objekt zu projizieren. Die präzise bekannte geometrische Beziehung zwischen Projektor und Kamera ermöglicht es, aus den aufgenommenen Daten exakte 3D-Informationen zu berechnen [37]. Hochwertige 3D-Scanner verwenden neben Laserprojektoren auch Projektoren, die mit Weiß- oder Infrarotlicht arbeiten, um flexibel auf verschiedene Umgebungen und Objektoberflächen reagieren zu können. Diese Technik bietet eine sehr hohe Genauigkeit bei der Erfassung von 3D-Daten, bringt jedoch signifikante Nachteile mit sich.
Ein wesentlicher Nachteil dieses Verfahrens ist die Notwendigkeit eines Projektors, der für die Projektion der Lichtmuster unerlässlich ist. Insbesondere Laserprojektoren sind kostspielig und schwer zu beschaffen, was die Wirtschaftlichkeit gemäß **HW-GE 1** einschränkt. Alternativ können herkömmliche Videoprojektoren verwendet werden, was jedoch die Mobilität des Systems beeinträchtigt, da solche Projektoren aufgrund ihrer Größe und ihres Gewichts unhandlich sind. Dies widerspricht der Anforderung an die Ergonomie (**HW-SE 3** und **HW-SE 4**) für ein handgeführtes System.
Obwohl es miniaturisierte Projektoren gibt, sind diese in der Regel sehr teuer und überschreiten das vorgegebene Budget. Zudem kann die erforderliche Stromversorgung solcher Geräte die Handhabung weiter erschweren. Darüber hinaus besteht bei laserbasierten Systemen ein erhöhtes Sicherheitsrisiko (**HW-GE 2**), da Laserstrahlen potenziell gesundheitsschädlich sein können. Insgesamt erfordern aktive Projektionsverfahren eine sorgfältige Abwägung zwischen Genauigkeit, Mobilität, Kosten und Sicherheit.

Ergebnis

Die Auswahl der optimalen Aufnahmemethode muss gemäß der Anforderung **HW-GE 1** innerhalb eines Preisrahmens von 250 € realisierbar sein. Während in anderen Veröffentlichungen, wie beispielsweise in [92], durch den Einsatz aktiver Projektionsverfahren und hochwertiger Komponenten hervorragende Ergebnisse hinsichtlich der Erfassungsgenauigkeit erzielt wurden, zeigt sich jedoch, dass diese Technologie für das vorliegende Projekt aufgrund der Budgetbeschränkungen nicht geeignet ist. Insbesondere in Bezug

Kriterium	Aktive Projektionsverfahren	Stereophotogrammetrie
Erfassungsgenauigkeit	**Sehr gut** - Hohe Präzision durch aktive Beleuchtung und Kontrolle der Projektionsmuster; besonders effektiv bei strukturlosen oder homogenen Oberflächen	**Gut bis mittel** - Gute Genauigkeit bei Objekten mit ausreichender Textur und gutem Kontrast; Einschränkungen bei strukturlosen oder glänzenden Oberflächen
Robustheit gegenüber Umgebungslicht	**Mittel** - Umgebungslicht kann die Projektion beeinflussen; durch Einsatz von Infrarotlicht oder Filtern kann die Störanfälligkeit reduziert werden	**Mittel** - Variable Lichtverhältnisse können die Bildqualität beeinträchtigen; ausreichende Beleuchtung erforderlich
Handhabung und Benutzerfreundlichkeit	**Mittel** - Erfordert sorgfältige Kalibrierung und Handhabung; Systemkomplexität durch zusätzlichen Projektor erhöht	**Gut** - Einfachere Systemkonfiguration mit weniger Komponenten; dennoch genaue Kalibrierung der Kameras erforderlich
Kosten und Verfügbarkeit	**Mittel bis hoch** - Spezialisierte Projektoren erhöhen die Kosten; preiswerte Alternativen möglich, aber mit Leistungseinbußen verbunden	**Gut** - Verwendung handelsüblicher Kameras reduziert Kosten; breite Verfügbarkeit von Hardware
Anfälligkeit für externe Störungen	**Mittel** - Partikel wie Staub oder Nebel können die Projektion beeinträchtigen; weniger anfällig bei Infrarotprojektion	**Mittel** - Schwierigkeiten bei spiegelnden oder transparenten Oberflächen; texturlose Bereiche sind problematisch
Einsatzbereiche und Flexibilität	**Gut** - Geeignet für vielfältige Objekte, insbesondere solche mit wenig Textur; hohe Präzision in kontrollierten Umgebungen	**Gut** - Flexibel einsetzbar in verschiedenen Umgebungen; ideal für Objekte mit detaillierter Textur
Ergonomie und Mobilität	**Mittel** - Zusätzlicher Projektor erhöht Größe und Gewicht (budgetgerecht – nicht miniaturisiert); kompakte Systeme möglich, aber dadurch meist teurer	**Sehr gut** - Kompaktes und leichtes Setup durch Verwendung geeigneter Kameras; ideal für handgeführte Systeme
Sicherheitsaspekte	**Gut bis mittel** - Bei Verwendung von Laserprojektoren besteht Risiko für Augen; Infrarot ist sicherer	**Sehr gut** - Keine besonderen Sicherheitsrisiken; nutzt passiv das Umgebungslicht

Tabelle 5.1: Vergleich aktive Projektionsverfahren mit Stereophotogrammetrie

auf die Anforderung **HW-SE 4** erweist sich eine ausreichende Miniaturisierung innerhalb des vorgegebenen Budgets als nicht realisierbar.
Nach einer Analyse der beiden Methoden zeigt sich, dass die Stereophotogrammetrie mehrere entscheidende Vorteile für den Einsatz in einem handgeführten 3D-Scanner bietet. Die Methode zeichnet sich durch geringere Kosten und eine höhere Mobilität aus, was besonders für Anwendungen mit einem begrenzten Budget und hohen Anforderungen an die Handhabung von Vorteil ist. Darüber hinaus ist die Implementierung der Stereophotogrammetrie einfacher und sicherer, da keine potenziell gefährlichen Lichtquellen wie Laser erforderlich sind.
In Anbetracht dieser Überlegungen wird die Stereophotogrammetrie als die geeignete Methode für die Entwicklung des handgeführten 3D-Scanners festgelegt. Diese Entscheidung basiert auf einer ausgewogenen Betrachtung von Kosten, Mobilität, Sicherheit und Erfassungsgenauigkeit, die alle wesentlichen Anforderungen des Projekts berücksichtigt.

5.1.2 Sensorauswahl

Der Markt bietet eine breite Palette von Tiefenkameras, die nach dem Prinzip der Stereophotogrammetrie arbeiten. Um ein geeignetes System für die vorliegende Anwendung auszuwählen, werden im Folgenden drei spezifische Modelle verglichen, die nach einer sorgfältigen Recherche der verfügbaren Produkte ausgewählt wurden.

Die Modellreihe D400 von Intel RealSense umfasst verschiedene Tiefenkameras, die für unterschiedliche Anwendungsbereiche konzipiert sind. Für diesen Vergleich wurden die Modelle **D415** und **D405** ausgewählt. Für frühere Arbeiten an der HAW Hamburg wurden Stereokameras vom Typ **Intel RealSense D415** angeschafft. Diese Kameras können für die vorliegende Arbeit erneut verwendet werden, wodurch keine zusätzlichen Kosten entstehen würden.
Die D415 wird vom Hersteller explizit für Anwendungen empfohlen, bei denen eine hohe Genauigkeit erforderlich ist. Die D405 verfügt über Eigenschaften, die sie für die geplante Anwendung ebenfalls geeignet erscheinen lassen. Beide Modelle verfügen neben den Stereokameras über einen RGB-Sensor, der Farbbilder der betrachteten Szene aufnehmen kann. Die D415 ist zudem mit einem Infrarotprojektor ausgestattet, der wenig texturierte Szenen mit einem Muster optisch strukturieren kann [20].
Um auch ein Modell eines anderen Herstellers in die Analyse einzubeziehen, wird die **ZED Mini** von StereoLabs untersucht. Dieses Modell bietet im Vergleich zu anderen Produkten des Herstellers die besten Voraussetzungen für ein handgeführtes Scansystem. Die ZED Mini verfügt über einen RGB-Sensor und eine Inertial Measurement Unit (IMU), mit der die Bewegung der Kamera im Raum ermittelt werden kann, womit der Anforderung **SW-DA 1** entsprochen werden kann [86].

Punktauflösung

Für die präzise Erfassung eines Objekts ist eine hohe Punktauflösung der Tiefenkamera entscheidend (**SW-GE 3**). Die Punktauflösung gibt an, wie viele Messpunkte in einem definierten Bereich des Objekts erfasst werden können, und hängt von der Sensorauflösung, dem minimalen Arbeitsabstand und dem Sichtfeld der Kamera ab. Im Folgenden wird die Punktauflösung basierend auf diesen Parametern, die den jeweiligen Datenblättern der Modelle entnommen wurden, berechnet und verglichen [20, 87, 85]. Es ist jedoch wichtig zu beachten, dass diese Berechnungen ausschließlich auf der Hardwarebeschreibung basieren. Aktuelle Algorithmen können bei der Korrespondenzsuche deutlich

höhere Auflösungen im Bereich von Subpixeln erzielen. Beispielsweise wird in [44] beschrieben, dass die von den RealSense-Kameras verwendeten Algorithmen zur Analyse der Parallaxenverschiebung mit Auflösungen im Bereich von 1/32 Pixel arbeiten. Dies ermöglicht deutlich höhere Punktauflösungen, als es die reine Hardwarebeschreibung vermuten lässt, und verbessert somit die Präzision der erfassten 3D-Daten erheblich.
Für die folgenden Berechnungen der Punktauflösungen werden sowohl der jeweilige minimale Arbeitsabstand als auch, um die Vergleichbarkeit zu gewährleisten, der größte minimale Arbeitsabstand aller Systeme verwendet.

Die **D415** verfügt über eine maximale Sensorauflösung von 1280 × 720 Pixeln bei einer Bildfrequenz von 30 Hz. Der minimale Arbeitsabstand beträgt 450 mm, welche dem größten minimalen Arbeitsabstand aller Systeme entspricht. Das horizontale Sichtfeld beträgt 65°, das vertikale 40°. Daraus ergibt sich beim minimalen Arbeitsabstand eine Punktdichte von etwa 4,84 Punkten pro mm^2.
Die Intel RealSense **D405** besitzt ebenfalls eine Auflösung von 1280 × 720 Pixeln bei 30 Hz, jedoch aufgrund der geringeren Baseline, dem Abstand zwischen den beiden Kameras, einen minimalen Arbeitsabstand von nur 100 mm. Das horizontale Sichtfeld beträgt 87°, das vertikale 58°. Basierend auf diesen Parametern ergibt sich bei dem minimalen Arbeitsabstand von 100 mm eine maximale Punktdichte von etwa 43,56 Punkten pro mm^2. Bei dem Vergleichsarbeitsabstand von 450 mm beträgt die Punktdichte etwa 2,25 Punkte pro mm^2.
Die **ZED Mini** bietet eine maximale Auflösung von 2208 × 1242 Pixeln bei 15 Hz und einen minimalen Arbeitsabstand von 100 mm. Mit einem horizontalen Sichtfeld von 73° und einem vertikalen von 46° ergibt sich bei dem minimalen Arbeitsabstand eine Punktdichte von etwa 218,59 Punkten pro mm^2. Bei dem Vergleichsarbeitsabstand von 450 mm beträgt die Punktdichte etwa 40,58 Punkte pro mm^2.

Die Bewertung der Punktdichte ist nicht trivial, da sie je nach Kamerasystem für unterschiedliche Abstände optimiert ist. Idealerweise sollte der Arbeitsabstand so gering wie möglich gewählt werden, um die höchstmögliche Genauigkeit der Daten zu erzielen. Dabei muss jedoch berücksichtigt werden, dass die erfasste Fläche bei größerem Abstand deutlich größer ist und somit eine größere Objektoberfläche aufgenommen werden kann. Beispielsweise kann mit der D405 beim minimalen Arbeitsabstand lediglich eine Fläche von 10 × 5cm erfasst werden, während sich diese bei einem Abstand von 450 mm auf etwa 42 × 25 cm vergrößert.
Unter Berücksichtigung dieser Aspekte und basierend auf der Anforderung **HW-AB 2** wird der Fokus auf einen Arbeitsabstand von 450 mm gelegt. Bei einem theoretischen

Objekt mit den Maßen 60 × 60 × 60 cm müssten mit der D405 rund 20-mal mehr Aufnahmen erfasst werden, als es bei dem größeren Arbeitsabstand der Fall wäre.

Tiefengenauigkeit und Tiefenrauschen

Die Tiefengenauigkeit ist ein wesentlicher Faktor bei der Erfassung von 3D-Daten, da sie die Präzision der Entfernungsbestimmung entlang der z-Achse angibt. Eine hohe Tiefengenauigkeit ist unerlässlich, um Verzerrungen und Abweichungen in den erfassten Tiefendaten zu minimieren und eine präzise Darstellung der Objektoberfläche zu gewährleisten. Die von den Herstellern angegebenen Werte für die Tiefengenauigkeit sind jedoch häufig nur grobe Schätzungen und bieten nicht immer eine ausreichende Grundlage für eine fundierte Beurteilung.
Zum Beispiel gibt Intel für die RealSense **D415** eine Tiefengenauigkeit von weniger als 2 % Abweichung bei einem Abstand von 2 Metern an, wobei der ideale Arbeitsabstand zwischen 0,5 und 3 Metern liegt. Für die RealSense **D405** wird eine Abweichung von weniger als 2 % bei einem Abstand von 50 cm angegeben, wobei der ideale Arbeitsabstand zwischen 7 und 50 cm liegt. Der minimale Abstand für die maximale Auflösung beträgt hier jedoch 100 mm. StereoLabs gibt für die **ZED Mini** eine Abweichung von weniger als 1 % bei Entfernungen von 2 m und weniger als 1,8 % bei Entfernungen bis 4 m an. Da diese Angaben keine präzise Einordnung der Leistungsfähigkeit ermöglichen, wird zusätzlich das Tiefenrauschen der jeweiligen Kameras als Vergleichsmaßstab herangezogen.
Tiefenrauschen ist ein kritischer Faktor in der 3D-Bildgebung, der die Qualität der erzeugten Tiefendaten stark beeinflussen kann. Es entsteht durch verschiedene Einflüsse, darunter die Eigenschaften des erfassten Objekts, wie Helligkeit und Oberflächenstruktur, die das Rauschen beeinflussen. Dies liegt daran, dass sichtbares Licht und Infrarotstrahlung von unterschiedlichen Materialien und Farben unterschiedlich absorbiert oder reflektiert werden. Beispielsweise reflektieren dunkle Oberflächen Infrarotstrahlen, wie sie vom Projektor der D415 ausgesendet werden, weniger gut. Externe Faktoren wie Umgebungslicht können ebenfalls das Tiefenrauschen verstärken. Sonnenlicht kann Bereiche überbelichten und so die Erfassung von Tiefendaten beeinträchtigen, während natürliche Infrarotstrahlung die relativ schwach projizierten Infrarotmuster überstrahlen kann.
Zusätzlich zu diesen externen Faktoren beeinflussen interne Parameter der Kamera wie Brennweite, Sichtfeld und die Baseline das Tiefenrauschen erheblich [47, 45], wobei die

Entfernung zum aufgenommenen Objekt den größten Einfluss hat.
Die Auswirkungen des Tiefenrauschens auf die erfassten Daten lassen sich anhand einer glatten Oberfläche verdeutlichen. Statt einer ideal glatten Fläche erscheinen kleine Höhen und Tiefen, die durch das Rauschen verursacht werden. Diese Störungen überlagern auch die Oberfläche anderer Objekte und führen zu Ungenauigkeiten in der resultierenden Punktwolke. Der durch das Tiefenrauschen bedingte Fehler kann mit den folgenden Formeln aus [45] berechnet werden:

$$DRMS = \frac{d^2 \cdot sp}{f \cdot b} \tag{5.1}$$

$$f = \frac{X_{res}}{2 \cdot tan(\frac{HFOV}{2})} \tag{5.2}$$

Dabei gilt:

- ***DRMS:*** Der Root Mean Square (RMS)-Fehler der Tiefenwerte in Millimetern, der bei der Aufnahme einer flachen Ebene mit gut texturierter Oberfläche in einer Entfernung ***d*** zum Sensor entsteht.
- ***b:*** Die Baseline in Millimetern.
- ***sp:*** Die Subpixelauflösung, die bei gut texturierten Objekten im Bereich von 1/10 Pixeln liegen kann.
- ***f:*** Die Brennweite in Pixeln, berechnet nach Gleichung (5.2).
- $\mathbf{X_{res}}$: Die horizontale Auflösung in Pixeln.
- ***HFOV:*** Das horizontale Sichtfeld.

Obwohl diese Methode nicht explizit für die Anwendung auf Tiefenkameras anderer Hersteller vorgesehen ist, wird sie hier zum Zweck der Vergleichbarkeit auch auf die ZED Mini angewendet. Da für die ZED Mini keine spezifischen Angaben zur Subpixelgenauigkeit vorliegen, wird konservativ von einer Subpixelauflösung von 1 ausgegangen. Um die Vergleichbarkeit zu gewährleisten, wird dieser Wert für alle Kameras verwendet. Der Vergleich wird wie zuvor bei der Punktauflösung bei einem Abstand von 450 mm

	D415		D405		ZED Mini	
	450 mm	**-**	**450 mm**	**100 mm**	**450 mm**	**100 mm**
Depth RMS Error	3,66 mm	-	15,83 mm	0,78 mm	2,15 mm	0,11 mm

Tabelle 5.2: Tiefenrauschen der betrachteten Kameras

durchgeführt. Zusätzlich werden zu Informationszwecken die Werte beim geringstmöglichen Abstand der anderen beiden Kameras berechnet. Die Ergebnisse des Vergleichs des Tiefenrauschens der drei Kameras sind in Tabelle 5.2 dargestellt.

Gewicht

Die Anforderung **HW-SE 6** legt fest, dass das Gesamtgewicht der Sensoreinheit 400 g nicht überschreiten darf. Ein geringes Gewicht der verwendeten Sensorik ist daher von großem Vorteil, da es eine großzügigere Dimensionierung der übrigen Komponenten der Sensoreinheit ermöglicht. In Tabelle 5.3 ist das Gewicht der analysierten Tiefenkameras dargestellt, um deren Eignung hinsichtlich dieser Anforderung zu bewerten. Ein leichterer Sensor kann den Bedienkomfort erheblich verbessern und die Handhabung des gesamten Systems erleichtern.

	D415	D405	ZED Mini
Gewicht	64 g	58 g	62,9 g

Tabelle 5.3: Gewicht der betrachteten Tiefenkameras

Schnittstellen

Die beiden RealSense-Kameras verfügen über eine USB-3.1-Schnittstelle, während die ZED Mini mit einer USB-3.0-Schnittstelle ausgestattet ist. In beiden Fällen ist eine ausreichend schnelle und zuverlässige Datenübertragung gewährleistet. Zur Steuerung und Integration der Hardware bieten beide Hersteller ein Software Development Kit (SDK) an, das Entwicklern ermöglicht, die Kameras in verschiedene Anwendungen zu integrieren und auf deren Einstellungen sowie Funktionen zuzugreifen.
Eine weitere Funktion, die sowohl die D415 als auch die ZED Mini unterstützen, ist die Hardwaresynchronisierung. Diese ermöglicht es, mehrere gleichartige Kameras synchron zu betreiben, was insbesondere dann von Vorteil ist, wenn eine höhere Datenrate oder ein erweitertes Sichtfeld erforderlich ist. Durch die Synchronisation mehrerer Kameras

können umfangreichere 3D-Daten simultan erfasst werden, wodurch die Flexibilität und Einsatzmöglichkeiten der Systeme erweitert werden.

Kosten

Die Kosten sind für diese Arbeit ein Kriterium, welches zwingend erfüllt werden muss. Nach Anforderung **HW-GE 1** liegt das Budget bei 250 €. In Tabelle 5.4 sind die Kosten zum Zeitpunkt der Erstellung der Arbeit für eine neuwertige Kamera angegeben, wenn diese direkt vom Hersteller beschafft werden.

	D415	**D405**	**ZED Mini**
Kosten	272,00 €	272,00 €	399,00 €

Tabelle 5.4: Kosten der betrachteten Tiefenkameras

Sonstige Eigenschaften

Alle betrachteten Kameras ermöglichen die Aufnahme von Farbbildern, wodurch die Textur des Objekts erfasst und in das 3D-Modell integriert werden kann. Es ist dabei zu beachten, dass die ZED Mini und die D405 zur Tiefenbestimmung ausschließlich im sichtbaren Lichtspektrum arbeiten, während die D415 zusätzlich den Infrarotbereich nutzt. Dies kann in bestimmten Szenarien von Vorteil sein, insbesondere bei schlechten Lichtverhältnissen, wenn das sichtbare Spektrum durch Umgebungsbedingungen beeinträchtigt wird, oder wenn eine untexturierte beziehungsweise sehr fein texturierte Szene erfasst werden soll.
Der Infrarotprojektor der D415 ist gemäß der Norm [63] als Laser der Klasse 1 klassifiziert. Dies bedeutet, dass der Laser in allen Anwendungen, selbst bei längerer Exposition, als sicher eingestuft wird und keine Gefahr für den Benutzer oder umstehende Personen darstellt.

Bewertung und Auswahl

Die Ergebnisse des Vergleichs dieser Systeme sind in Tabelle 5.5 zusammengefasst. Die Bewertung erfolgt anhand der folgenden Notation: Ein „+“ kennzeichnet ein positives Ergebnis, ein „-“ ein negatives Ergebnis, und ein „o“ steht für ein neutrales Ergebnis. Wenn ein direkter Vergleich der Leistungswerte im Sinne von besser oder schlechter

möglich ist, spiegelt diese Notation eine Rangfolge wider, wobei „o“ die mittlere Platzierung repräsentiert.
Basierend auf dem Vergleich erweist sich die **ZED Mini** technisch als die beste Option. Die Kamera bietet eine überlegene Punktauflösung und weist aufgrund der großen Baseline und hohen Auflösung eine sehr geringe Abweichung durch Tiefenrauschen auf. Dies deutet darauf hin, dass mit dieser Kamera die genauesten Punktwolken erfasst werden können.
Allerdings überschreiten alle betrachteten Kameras das vorgegebene Budget, sodass keine dieser Optionen im Rahmen dieser Arbeit beschafft werden kann. Daneben existieren kostengünstigere Stereokameramodule, die für den Einsatz mit Einplatinencomputern wie dem Raspberry Pi ausgelegt sind [11]. Diese Module bieten eine preiswerte Hardwarebasis für den Aufbau eines geeigneten Systems; jedoch lässt sich die erreichbare Genauigkeit eines solchen Systems nicht abschätzen. Aufgrund der Komplexität und des erforderlichen Aufwands beim Aufbau eines solchen Sensorsystems wird in dieser Arbeit darauf verzichtet. Aufgrund der Verfügbarkeit der D415 und ihrer technischen Eigenschaften, in denen sie sich gegenüber der D405 behaupten kann, wird dieses Kamerasystem für die vorliegende Arbeit ausgewählt.

	D415	**D405**	**ZED Mini**
Punktauflösung	o	-	+
Tiefenrauschen	o	-	+
Gewicht	-	+	o
Schnittstellen	+	o	+
Kosten	o	o	-
Sonstige Eigenschaften	+	o	o

Tabelle 5.5: Bewertung der betrachteten Kameras

5.1.3 Risikobetrachtung

Im Rahmen des festgelegten Hardwarekonzepts wurden keine Komponenten identifiziert, die eine erhöhte Gefährdung darstellen könnten. Eine potenzielle Gefährdung ergibt sich jedoch durch die kabelgebundene Anbindung der Sensoreinheit, welche eine Stolpergefahr für den Benutzer oder umstehende Personen darstellen kann. Diese Gefahr könnte durch den Einsatz einer kabellosen Lösung verringert werden, allerdings wird diese Option aufgrund der im Kapitel 4.4 festgelegten Budgetbeschränkungen (siehe Anforderung **HW-GE 1**) im Rahmen dieser Arbeit nicht weiterverfolgt. Stattdessen

wird die Stolpergefahr minimiert, indem der Scanvorgang ausschließlich außerhalb von Laufwegen durchgeführt wird. Zusätzlich können Warnhinweise angebracht werden, um auf die mögliche Stolpergefahr aufmerksam zu machen.
Die Wahl der Sensorik, insbesondere die Verwendung eines Lasers der Klasse 1, stellt sicher, dass keine zusätzlichen Schutzmaßnahmen, wie beispielsweise das Tragen von Schutzbrillen, erforderlich sind. Gemäß Anforderung **HW-GE 2** muss das System sicher sein und darf keine Gefahr für Personen darstellen. Da Laser der Klasse 1 in allen Anwendungen als sicher eingestuft werden, ist die Notwendigkeit für besondere Schutzvorkehrungen minimiert. Allgemeine Sicherheitsregeln, die bei der Verwendung solcher Lasersysteme zu beachten sind, werden in die abschließende Benutzeranweisung integriert, um einen sicheren Betrieb zu gewährleisten.

5.2 Sensoreinheit

Die Sensoreinheit integriert die verwendete Sensorik sowie die erforderliche Peripherie und ist so auszulegen, dass eine schnelle und präzise Datenerfassung ermöglicht wird. Um die Effizienz der Datenerfassung zu maximieren, werden beide verfügbaren D415-Kameras parallel eingesetzt. Dank der integrierten Synchronisationsfunktion der D415-Kameras können die Bilder beider Kameras gleichzeitig aufgenommen werden. Dadurch werden Ungenauigkeiten, die durch zeitlich versetzte Aufnahmen entstehen könnten, weitestgehend vermieden.
Um die Sensoreinheit gemäß der Anforderung **HW-SE 3** um das zu untersuchende Objekt führen zu können, wird eine Halterung entworfen und angefertigt, an der beide Kameras befestigt werden können. Diese Halterung ermöglicht eine stabile und präzise Führung der Kameras während des Scanvorgangs.
Damit das Gesamtgewicht der Sensoreinheit, das in der Anforderung **HW-SE 4** auf maximal 400 Gramm festgelegt ist, nicht überschritten wird, wird die Halterung mittels 3D-Druck gefertigt. Die Druckparameter werden so eingestellt, dass die Halterung ein Gewicht von 284 Gramm (400 Gramm abzüglich des Gewichts zweier Kameras) nicht überschreitet. Zudem wird die Positionierung und Ausrichtung der Kameras zueinander durch den Einsatz von austauschbaren Kamerahalterungen flexibel gestaltet, sodass die Sensoreinheit an verschiedene Anwendungsanforderungen angepasst werden kann.
Auf die Integration zusätzlicher Peripherie, wie in den Anforderungen **HW-SE 6** und **HW-SE 7** beschrieben, wird im ersten Prototyp verzichtet. Der Schwerpunkt liegt

zunächst auf der Implementierung der notwendigen Software zur Verarbeitung der erfassten Daten, um eine funktionale und effiziente Basis für die Weiterentwicklung des Systems zu schaffen.

5.3 Entwicklungsumgebung

Entscheidend für die Auswahl der Entwicklungsumgebung in dieser Arbeit ist die Unterstützung des Intel RealSense SDK, das den Zugriff auf die zuvor ausgewählte Tiefenkamera D415 ermöglicht. Laut den Angaben des Herstellers unterstützt das SDK die Entwicklung in den Programmiersprachen C/C++, C#, Matlab und Python. Aufgrund der Vorkenntnisse des Entwicklers dieser Arbeit wird die Entscheidung auf eine Auswahl zwischen C++ und Python eingegrenzt.
Im Folgenden werden die relevanten Aspekte der beiden Programmiersprachen beleuchtet, die bei der Auswahl berücksichtigt werden, um eine fundierte Entscheidung zu treffen. Dabei werden Kriterien wie Performance, Entwicklungsaufwand, Flexibilität und Integration mit anderen Bibliotheken und Werkzeugen betrachtet. Diese Analyse dient als Grundlage für die endgültige Entscheidung, welche Programmiersprache für die Entwicklung der Softwarekomponenten verwendet wird.

5.3.1 Entwicklungsaufwand

Python zeichnet sich durch eine klare und einfache Syntax aus, die die Entwicklung und Wartung des Codes erheblich vereinfacht. Im Gegensatz zu C++, das aufgrund seiner umfangreichen und komplexen Sprachfeatures eine steilere Lernkurve aufweist, ermöglicht Python eine schnellere Implementierung und eine leichtere Fehlerbeseitigung. Dies ist besonders vorteilhaft in Projekten, die komplexe Algorithmen und umfangreiche Datenverarbeitungsaufgaben umfassen, da die Entwicklungszeit signifikant reduziert werden kann. Darüber hinaus fördert Python durch seine hohe Lesbarkeit die Zusammenarbeit mit betreuenden Personen und erleichtert die spätere Anpassung und Erweiterung des Codes, was bei langfristigen Projekten einen wesentlichen Vorteil darstellt.

5.3.2 Bibliotheksvielfalt und -verfügbarkeit

Obwohl das Intel RealSense SDK sowohl für Python als auch für C++ verfügbar ist, bietet Python eine weitaus größere Vielfalt an zusätzlichen Bibliotheken, die für die Entwicklung und Integration von 3D-Scan-Lösungen von großem Nutzen sind. Bibliotheken wie *NumPy* und *SciPy* für numerische Berechnungen, *Matplotlib* und *Seaborn* für die Datenvisualisierung sowie *Pandas* für die Datenverarbeitung schaffen eine umfassende und flexible Entwicklungsumgebung. Diese breite Verfügbarkeit an Bibliotheken erleichtert nicht nur die Implementierung, sondern auch die Erweiterung des Systems und unterstützt somit die schnelle Entwicklung komplexer Funktionen.

5.3.3 Community und Support

Python verfügt über eine große und aktive Community, die eine Vielzahl von Ressourcen, Dokumentationen und Unterstützung bereitstellt. Dies vereinfacht die Lösung von Problemen und fördert die Anwendung bewährter Methoden in der Softwareentwicklung. Obwohl auch C++ eine umfangreiche Community hat, sind die Ressourcen für Python aufgrund seiner zunehmenden Beliebtheit und seiner einfacheren Handhabung oft zugänglicher und umfassender.

5.3.4 Performance

Ein wesentlicher Vorteil von C++ gegenüber Python liegt in der Ausführungsgeschwindigkeit. Als kompilierte Sprache, die direkt in maschinenlesbaren Code übersetzt wird, kann C++ in der Regel schneller ausgeführt werden. Daher sind rechenintensive Funktionen in Bibliotheken wie *OpenCV* und der *Intel RealSense SDK* häufig als kompilierte Programme implementiert, die in C++ geschrieben sind. Dies bedeutet, dass bei der Entwicklung mit Python zwar die Flexibilität und Einfachheit der Sprache genutzt werden kann, jedoch müssen Änderungen an diesen leistungsintensiven Funktionen direkt im C++-Quellcode vorgenommen werden, was zusätzliche Komplexität und Aufwand mit sich bringen kann.

5.3.5 Entscheidung

Zusammenfassend lässt sich festhalten, dass Python aufgrund seiner einfachen Syntax, der Verfügbarkeit einer Vielzahl leistungsstarker Bibliotheken und der Unterstützung durch eine große Community die geeignetere Wahl für die Entwicklung des handgeführten 3D-Scansystems darstellt. Obwohl C++ in Bezug auf die Ausführungsgeschwindigkeit Vorteile bietet, überwiegen die genannten Vorteile von Python, insbesondere wenn optimierte Bibliotheken genutzt werden, die für dieses Projekt relevant sind.
Sollte die Notwendigkeit bestehen, bestimmte Funktionen zugunsten einer höheren Ausführungsgeschwindigkeit auszulagern, können diese in C++ implementiert, kompiliert und dann über das Python-Programm aufgerufen werden. Dies ermöglicht es, die Stärken beider Sprachen zu kombinieren und ein optimales Gleichgewicht zwischen Entwicklungsaufwand und Performance zu erreichen.

5.4 Vorgehen

Auf Grundlage der getroffenen Entscheidungen kann nun ein detailliertes Konzept für die Softwareentwicklung ausgearbeitet werden, mit dem die Tiefendaten so erfasst und verarbeitet werden, dass alle gestellten Anforderungen erfüllt werden. Im Folgenden wird zunächst systematisch beschrieben, welche einzelnen Schritte notwendig sind, wie diese ausgeführt werden und in welcher Reihenfolge sie erfolgen müssen, um eine effiziente und präzise Datenerfassung und -verarbeitung zu gewährleisten.

5.4.1 Verwendung von Bibliotheken und Frameworks

Um die Entwicklung effizient zu gestalten, werden etablierte Bibliotheken und Frameworks eingesetzt. Für die zweidimensionale Bildverarbeitung wird hauptsächlich *OpenCV* verwendet, das umfangreiche Funktionen für Bildverarbeitung und Kalibrierungsmethoden bietet. Die Datenerhebung und Kommunikation mit der Sensorik erfolgt über das *Intel RealSense SDK*.
Für die Verarbeitung der 3D-Daten kommen je nach Anwendungsfall *Open3D*, die *Point Cloud Library (PCL)* oder *SciPy* zum Einsatz. Diese Bibliotheken überschneiden sich in manchen Bereichen in ihren Funktionen, wobei sie je nach Bibliothek unterschiedlich

und teilweise auch effizienter implementiert sind. Die Interoperabilität dieser Bibliotheken wird durch die Verwendung standardisierter Datenstrukturen gewährleistet.
Python bietet eine Vielzahl von SDKs zur Erstellung eines GUI. An dieser Stelle wird jedoch kein detaillierter Vergleich der verschiedenen Lösungen vorgenommen. Für die Benutzerschnittstelle bestehen keine spezifischen Anforderungen an ein bestimmtes Framework, da die erwarteten Funktionen mit den meisten gängigen Frameworks für Python realisiert werden können. Da die Entwicklung der Benutzeroberfläche nicht im Mittelpunkt dieser Arbeit steht, sondern vielmehr ein Mittel zum Zweck darstellt, wurde die Entscheidung getroffen, PyQt zu verwenden.
PyQt wurde ausgewählt, weil es dem Ersteller dieser Arbeit aus vorherigen Projekten vertraut ist, was eine effiziente Entwicklung begünstigt. PyQt basiert auf dem Qt-Framework und bietet eine breite Palette an Widgets sowie umfangreiche Funktionen für die Erstellung komplexer GUIs. Obwohl die Lernkurve bei PyQt steiler ist als bei einigen anderen Python-GUI-Frameworks, bietet es den Vorteil des *Qt Designers*. Der *Qt Designer* ist ein Tool, das es ermöglicht, das Layout der GUI mit Drag-and-Drop-Methoden visuell zu erstellen. Nachdem das Layout zusammengestellt wurde, können die entsprechenden Funktionalitäten programmiert und mit den GUI-Elementen verbunden werden.
PyQt wird mit einem dualen Lizenzsystem vertrieben: einer kommerziellen Lizenz, die erworben werden muss, wenn die andere Lizenz nicht angewandt werden kann oder soll, und der GNU General Public License Version 3 (GPLv3) [19]. Unter der GPLv3 kann die Software kostenfrei genutzt werden, vorausgesetzt, dass den Anwendern der entwickelten Software der vollständige Quellcode zur Verfügung gestellt wird und dieser ebenfalls unter der GPLv3 veröffentlicht wird. Da diese Arbeit über die HAW-Hamburg veröffentlicht wird, kann diese Lizenz angewandt werden, und die Software kostenfrei genutzt werden. Die Software wird interessierten Nutzern inklusive des Quellcodes über die Hochschule zur Verfügung gestellt werden. Die notwendigen Schritte zur Verwendung der Lizenz sind in [33] beschrieben.

5.4.2 Gesamtablauf

In Abbildung 5.1 ist der Gesamtablauf eines Aufnahmeprozesses zur Rekonstruktion eines Objekts als Aktivitätsdiagramm dargestellt. Dieses Diagramm visualisiert die Abfolge der notwendigen Schritte von der Hardwareeinrichtung über die Datenaufnahme bis hin zur finalen Rekonstruktion. In den folgenden Abschnitten werden diese Schritte

im einzelnen beschrieben.
Im darauf folgenden Kapitel wird die praktische Umsetzung der einzelnen Schritte entwickelt und detailliert beschrieben. Das Diagramm dient dabei als Leitfaden, um sicherzustellen, dass alle erforderlichen Prozesse systematisch und in der richtigen Reihenfolge ausgeführt werden, um eine erfolgreiche 3D-Rekonstruktion des Objekts zu gewährleisten.

5.4.3 Prozess konfigurieren

Bevor die Datenaufnahme beginnen kann, muss der Prozess der Datenerfassung konfiguriert werden. In diesem Schritt wird festgelegt, wie viele Einzelpunktwolken aufgenommen werden sollen, um das gesamte Objekt oder die Szene vollständig zu erfassen. Die Anzahl der benötigten Punktwolken hängt vor allem von der Komplexität und der Größe des Objekts ab. Ein komplexeres oder größeres Objekt erfordert in der Regel mehr Aufnahmen, um alle Details ausreichend zu erfassen und Lücken in der Punktwolke zu vermeiden.
Neben der Festlegung der Anzahl der Punktwolken können auch weitere Einstellungen vorgenommen werden, die den Aufnahmeprozess maßgeblich beeinflussen. Dazu gehört beispielsweise die Auswahl spezifischer Filter, die während der Aufnahme angewendet werden sollen, um die Qualität der erfassten Daten zu optimieren.
Darüber hinaus werden an dieser Stelle alle weiteren Konfigurationen und Einstellmöglichkeiten vom Benutzer vorgenommen. Diese umfassen beispielsweise die Wahl der Auflösung und der Bildrate für den folgenden Schritt. An diesem Punkt wird aufgrund der geringen Komplexität des Schrittes auf ein separates Ablaufdiagramm verzichtet.

5.4.4 Einrichtung der Hardware

Bevor die Datenaufnahme beginnen kann, muss die Hardware vollständig eingerichtet und funktionsfähig sein. Dieser Schritt beinhaltet den Aufbau der Hardwareverbindung sowie die Initialisierung. Die folgenden Schritte müssen dabei durchgeführt werden:

1. Kommunikation mit der Hardware herstellen

2. Konfigurieren der Hardware

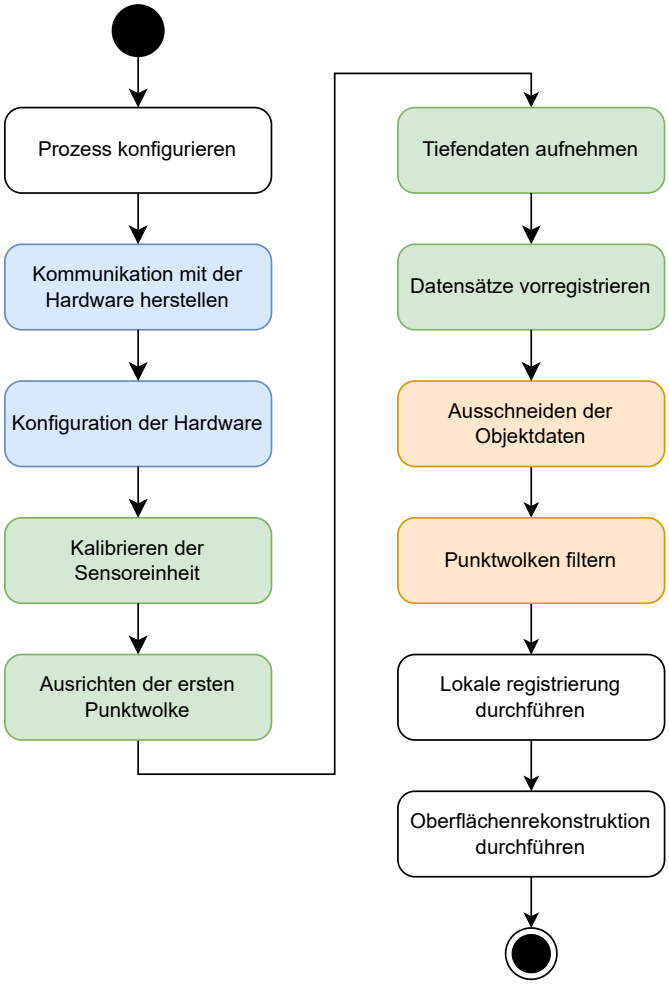


Abbildung 5.1: Aktivitätsdiagramm des gesamten Prozesses zur Objektrekonstruktion. Die blauen Boxen repräsentieren die Schritte zur Einrichtung der Hardware, die grünen Boxen stehen für die Phase der Datenaufnahme, und die orangenen Boxen zeigen die Schritte der Vorverarbeitung der Punktwolken.

Kommunikation mit der Hardware herstellen

Zunächst muss sichergestellt werden, dass die verwendeten Kameras und alle relevanten Hardwarekomponenten korrekt mit dem Computersystem verbunden und ansprechbar

sind. Im Fall der Kameras erfolgt die Kommunikation über das Intel RealSense SDK, das eine Schnittstelle zur Steuerung und Abfrage der Kameras bereitstellt. Der Datenfluss innerhalb des Systems basiert auf sogenannten Pipelines, die für jede Kamera separat erstellt und konfiguriert werden müssen.
Zu Beginn der Initialisierung muss überprüft werden, ob tatsächlich zwei Kameras angeschlossen und korrekt erkannt wurden. Dies ist entscheidend, um sicherzustellen, dass beide Kameras verfügbar sind und synchron betrieben werden können.
Bei der Einrichtung der Pipelines muss spezifiziert werden, welche Datenströme (zum Beispiel Tiefenbilder, Infrarotbilder oder Farbbilder) genutzt werden sollen. Zudem müssen die gewünschten Auflösungen und Datenraten für die einzelnen Streams festgelegt werden. Nach [45] ist die beste Tiefengenauigkeit mit der höchstmöglichen Auflösung zu erreichen, weshalb diese auf 1290 x 720 Pixel festgelegt wird. Die Bildrate wird zunächst auf sechs Bilder pro Sekunde festgelegt, da dies die geringste verfügbare Einstellung ist. Da die Bildaufnahme zunächst manuell ausgelöst werden wird, ist keine hohe Bildrate erforderlich.

Konfigurieren der Hardware

Die Konfiguration der Hardware beeinflusst maßgeblich die Qualität der erzeugten Daten. Die RealSense SDK bietet eine Vielzahl von Parametern, die in diesem Schritt eingestellt beziehungsweise übernommen werden müssen, um die Leistungsfähigkeit der Kameras voll auszuschöpfen. Die Einstellung dieser Parameter kann komplex sein, wird jedoch durch die Verwendung von Presets erleichtert, die vom Hersteller für verschiedene Anwendungsfälle bereitgestellt werden. Diese Presets bieten vorab optimierte Einstellungen, die auf typische Szenarien zugeschnitten sind und somit eine solide Ausgangsbasis für die Konfiguration darstellen.

In Abbildung 5.2 ist das Aktivitätsdiagramm zur Einrichtung der Hardware dargestellt, welches beide zuvor beschriebenen Schritte inkludiert.

5.4.5 Datenaufnahme

Der für die weitere Verarbeitung der Daten wichtigste Schritt ist die Datenaufnahme. Mit der ausgewählten Sensorik muss ein Ablauf entwickelt werden, mit dem Tiefendaten derart erfasst werden können, dass diese im Anschluss für die weitere Verarbeitung vorbereitet sind. Folgende Funktionen müssen dafür implementiert werden:

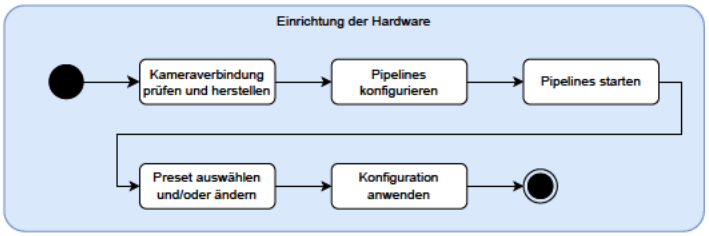


Abbildung 5.2: Aktivitätsdiagramm zur Einrichtung der Hardware

1. Tiefendaten aufnehmen
2. Kalibrieren der zwei Kameras der Sensoreinheit zueinander
3. Datensätze aufnehmen

Im Folgenden werden diese Schritte detailliert beschrieben, wobei insbesondere auf die Gründe eingegangen wird, warum sie durchgeführt werden. Die Wahl der spezifischen Methodiken und ihre Umsetzung werden anschließend im Kapitel zur praktischen Umsetzung näher erläutert.

Tiefendaten aufnehmen

Mit der zuvor konfigurierten Hardware werden nun die Tiefen- und Farbbilder aufgenommen. Für jede Kamera ergibt sich als Ergebnis der Aufnahme ein Tiefenbild sowie ein Farbbild. Das Tiefenbild enthält für jeden Pixel einen Wert in Metern, der den Abstand des entsprechenden Punktes in der realen Welt zum Ursprung des Kamerakoordinatensystems beschreibt. Bei ungültigen Tiefenwerten, die beispielsweise durch einen zu geringen oder zu großen Objektabstand entstehen, wird der Wert auf null gesetzt. Das Farbbild speichert für jeden Punkt im Tiefenbild einen entsprechenden RGB-Farbwert, der die visuelle Textur des aufgenommenen Objekts darstellt.
Beim Aufnehmen der Bilder muss sichergestellt werden, dass die Aufnahmen beider Kameras im gleichen Format vorliegen, um eine problemlose Weiterverarbeitung und Fusion der Daten zu gewährleisten. Darüber hinaus ist es wichtig, dass die Farbbilder korrekt auf die jeweiligen Tiefenbilder ausgerichtet sind. Diese Ausrichtung stellt sicher, dass jeder Tiefenwert mit dem korrekten Farbwert verknüpft wird

Kalibrieren der zwei Kameras der Sensoreinheit zueinander

Um aus den beiden Teilaufnahmen einen zusammenhängenden Datensatz zu erstellen, müssen die beiden Kameras zueinander kalibriert werden. Jede der Kameras erzeugt Tiefenbilder, die sich an ihrem internen Koordinatensystem orientieren, wobei der Ursprung dieses Systems im Fokuspunkt der linken Stereokamera liegt (aus der Sicht der Kamera betrachtet). Die Z-Achse der Tiefendaten verläuft dabei parallel zur Sichtachse dieser Kamera. Um die beiden Aufnahmen in einer gemeinsamen Punktwolke mit einem einheitlichen Koordinatensystem zu vereinen, ist eine Kalibrierung der Kameras zueinander erforderlich.
Für die Kalibrierung wird zunächst die räumliche Beziehung zwischen den beiden Kameras bestimmt. Dies umfasst die Berechnung einer Transformationsmatrix, die beschreibt, wie die Koordinaten eines Punktes von einem Koordinatensystem in das andere überführt werden können. Häufig wird dafür ein Kalibrierobjekt, wie zum Beispiel ein Schachbrettmuster, verwendet, das von beiden Kameras gleichzeitig erfasst wird. Durch die Analyse der Position des Kalibrierobjekts in den erzeugten Daten können die notwendigen Transformationsparameter berechnet werden.
Die Transformation zwischen den beiden Koordinatensystemen wird durch eine Kombination aus Rotation und Translation beschrieben. Die Rotationsmatrix gibt an, wie das Koordinatensystem der zweiten Kamera gedreht werden muss, um es mit dem der ersten Kamera in Übereinstimmung zu bringen. Die Translationsmatrix beschreibt die räumliche Verschiebung zwischen den beiden Kameras.
Sobald die Transformationsmatrix bestimmt wurde, kann sie auf die Punktwolken der zweiten Kamera angewendet werden, sodass die Punkte dieser Kamera in das Koordinatensystem der ersten Kamera umgerechnet werden. Das Ergebnis ist eine vereinte Punktwolke, in der alle Datenpunkte beider Kameras in einem gemeinsamen Koordinatensystem vorliegen. Dieser Prozess kann entweder im Voraus, statisch, oder zur Laufzeit anhand der aufgenommenen Teildaten dynamisch durchgeführt werden.
Darüber hinaus kann die Kalibrierung mit Hilfe der zweidimensionalen Bilddaten als auch mit den dreidimensionalen Punktwolken durchgeführt werden. Die Wahl des Ansatzes hängt von den spezifischen Anforderungen der Anwendung ab und beeinflusst die Genauigkeit und Effizienz der resultierenden Datenfusion.

Datensätze aufnehmen

Der erste aufgenommene Datensatz dient als Referenz für alle nachfolgenden Datensätze und wird daher auch bei der späteren Feinregistrierung als Ursprung angenommen. Um die anschließenden Schritte der Datenverarbeitung und Registrierung zu vereinfachen, wird diese erste Punktwolke so ausgerichtet, dass ihre Unterseite auf der XY-Ebene bei $Z = 0$ liegt. Die Unterseite der Punktwolke wird dabei als die Fläche definiert, mit der das beobachtete Objekt im Aufnahmebereich aufliegt. Diese Ausrichtung gewährleistet eine konsistente Bezugsebene für alle weiteren Punktwolken, die in das gemeinsame Koordinatensystem integriert werden. Die vom Anwender gewünschte Anzahl von Datensätze wird anschließend in einer Schleife aufgenommen. Zwischen den Aufnahmen wird die Sensorik vom Anwender in eine neue Position bewegt.

Datensätze Vorregistrieren

Der abschließende Schritt der Datenaufnahme umfasst die Durchführung einer groben Registrierung, auch bekannt als globale Registrierung. In diesem Schritt werden die Punktwolken der separat aufgenommenen Ansichten zusammengeführt und damit eine erste Ausrichtung der Datensätze vorgenommen.
Es gibt verschiedene Methoden zur globalen Registrierung, die jeweils unterschiedliche Algorithmen und Ansätze verwenden, um eine erste grobe Übereinstimmung der Punktwolken zu erzielen. Ziel dieser globalen Registrierung ist es, die Punktwolken so auszurichten, dass sie sich **dem globalen Minimum annähern**. Das globale Minimum repräsentiert den Zustand, in dem die Punktwolken bestmöglich zueinander ausgerichtet sind.
Nach der globalen Registrierung folgt typischerweise eine Feinregistrierung, bei der das lokale Minimum gesucht wird, welches im besten Fall dem globalen Minimum entspricht. Diese Feinregistrierung optimiert die Ausrichtung der Punktwolken weiter, indem sie kleinere Abweichungen korrigiert und sicherstellt, dass die Punktwolken präzise übereinandergelegt werden. Diese beiden Schritte sind entscheidend für die Erstellung einer konsistenten und akkuraten Gesamtdarstellung des erfassten Objekts.

In Abbildung 5.3 ist das detaillierte Aktivitätsdiagramm zur Datenaufnahme dargestellt, welches die zuvor beschriebenen Schritte inkludiert.

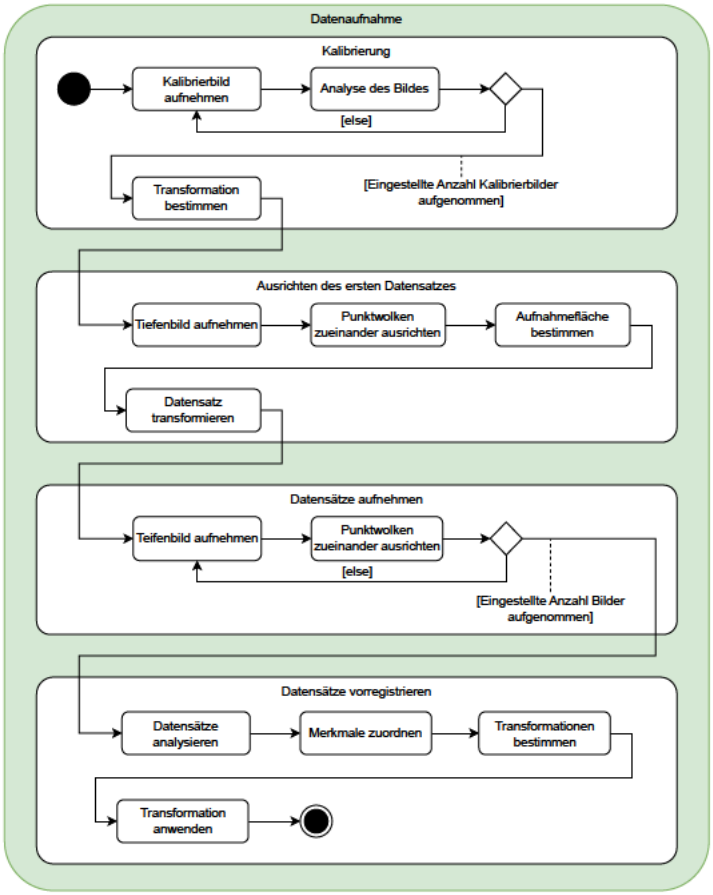


Abbildung 5.3: Aktivitätsdiagramm zur Datenaufnahme

5.4.6 Vorverarbeitung

Nachdem die Datensätze für einen kompletten Scan aufgenommen wurden, folgt die Vorverarbeitung, die sicherstellt, dass die Punktwolken bei der anschließenden Feinregistrierung korrekt zusammengefügt werden können. Dieser Schritt besteht aus den folgenden Methoden, die implementiert werden müssen.

1. Ausschneiden des Objekts

2. Filtern der Punktwolken

Im Folgenden werden die einzelnen Schritte der Vorverarbeitung beschrieben. Die Beschreibung und der Vergleich der verschiedenen Methodiken wird wie zuvor bei der Datenaufnahme im Umsetzungskapitel vorgenommen.

Ausschneiden des Objekts

Die aufgenommenen Punktwolken bestehen nach der Aufnahme aus einer Mischung von Punkten, die dem Hintergrund, dem Aufnahmebereich und dem beobachteten Objekt zugeordnet werden können. Um die Effizienz der weiteren Verarbeitungsschritte zu steigern, ist es sinnvoll, die Punkte, die dem beobachteten Objekt zugeordnet werden können, zu isolieren und die restlichen Punkte auszuschneiden. Durch das Entfernen der nicht relevanten Punkte, die dem Hintergrund oder dem Aufnahmebereich zugeordnet sind, kann die Gesamtzahl der zu verarbeitenden Punkte deutlich reduziert werden.
Da die Punktwolken alle auf der XY-Ebene ausgerichtet sind, lässt sich der Aufnahmebereich gut identifizieren und präzise entfernen. Diese Vorverarbeitung führt zu einer erheblichen Reduktion des Datenvolumens, wodurch die darauf folgenden Schritte, wie die Feinregistrierung und weitere Analysen, effizienter ausgeführt werden können. Durch die Konzentration auf die relevanten Objektpunkte wird die Rechenlast verringert, was sowohl die Geschwindigkeit als auch die Genauigkeit der nachfolgenden Algorithmen verbessert.

Filtern der Punktwolken

Am Ende der Vorverarbeitung werden die Punktwolken gefiltert, um die Datenqualität weiter zu verbessern. Dieser Schritt dient dazu, verbleibende Ausreißer zu entfernen, die sich nicht eindeutig dem Objekt zuordnen lassen und bei der vorherigen Verarbeitung nicht eliminiert wurden. Solche Ausreißer können durch unerwünschte Reflexionen, Sensorfehler oder ungünstige Aufnahmebedingungen entstanden sein.
Zusätzlich können Bereiche der Punktwolke entfernt werden, die eine geringere Punktdichte aufweisen als andere, was häufig bei Flächen auftritt, die in einem zu flachen Winkel zur Kamera aufgenommen wurden. Diese dünn besetzten Bereiche können zu Ungenauigkeiten in der späteren Datenverarbeitung führen und werden daher im Filterungsprozess gezielt ausgesondert.
Falls es als notwendig erachtet wird, kann in diesem Schritt auch ein temporales Filter

implementiert werden. Dieses Filter ermöglicht es, mehrere aufeinanderfolgende Aufnahmen zu mitteln, wodurch das Rauschen in den Punktwolken reduziert und die Genauigkeit weiter gesteigert wird. Diese Mittelung minimiert zufällige Fehler oder Rauschen, die in einzelnen Aufnahmen auftreten können, und führt zu einer klareren und konsistenteren Darstellung des erfassten Objekts.
Durch diese Filterung wird sichergestellt, dass die verbleibenden Punktwolken von guter Qualität sind, was eine präzise und zuverlässige Grundlage für die nachfolgende Weiterverarbeitung und Analyse bildet.

In Abbildung 5.4 ist das Aktivitätsdiagramm zur Vorverarbeitung der Daten dargestellt, welches die zuvor beschriebenen Schritte visualisiert.

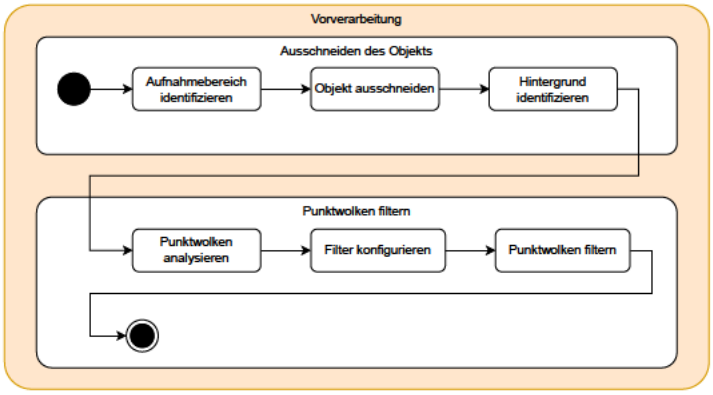


Abbildung 5.4: Aktivitätsdiagramm zur Vorverarbeitung

5.4.7 Lokale Registrierung

Bei der lokalen Registrierung werden die erzeugten Punktwolken final ausgerichtet, um eine zusammenhängende und kohärente Punktwolke zu erzeugen. Dieser Prozess ist entscheidend, um sicherzustellen, dass die einzelnen Punktwolken exakt übereinanderliegen und eine präzise Rekonstruktion des erfassten Objekts ermöglichen.
Es existieren zahlreiche Methoden zur lokalen Registrierung, und es ist wichtig, den Ansatz zu ermitteln, der für die vorliegenden Daten die genauesten Ergebnisse liefert. Die Wahl der Methode hängt von verschiedenen Faktoren ab, wie der Beschaffenheit der Punktwolken, der Komplexität der Szenen und Anforderungen an die Genauigkeit.

Viele dieser Methoden zur Feinregistrierung sind jedoch sehr rechenintensiv und erfordern daher eine gezielte Anwendung. Eine effektive Vorregistrierung ist dabei unerlässlich, um sicherzustellen, dass die Punktwolken bereits nah an ihrer optimalen Position zueinander liegen, bevor die Feinregistrierung erfolgt. Dies minimiert den Rechenaufwand und erhöht die Wahrscheinlichkeit, das globale Minimum der Fehlerfunktion zu erreichen.
Durch eine sorgfältige Auswahl und Anwendung der Feinregistrierungsmethoden kann die Qualität der finalen Punktwolke erheblich verbessert werden, was letztlich zu einer präzisen und detaillierten 3D-Rekonstruktion führt. Da die lokale Registrierung in einem Ablaufschritt stattfindet, wird an dieser Stelle auf ein Ablaufdiagramm verzichtet.

5.4.8 Oberflächenrekonstruktion

Der letzte Schritt der Verarbeitungskette ist die Oberflächenrekonstruktion. In diesem Schritt wird aus der zuvor generierten Punktwolke eine zusammenhängende Oberfläche erstellt und als 3D-Objekt gespeichert. Die Oberflächenrekonstruktion ist entscheidend für die Erstellung eines physisch interpretierbaren Modells, das für Visualisierungen, Simulationen oder die Fertigung genutzt werden kann.
Auch bei der Oberflächenrekonstruktion stehen verschiedene Algorithmen zur Verfügung, die jeweils unterschiedliche Eigenschaften der Punktwolke berücksichtigen und verschiedene Ergebnisse bei den erstellten Oberflächen liefern. Einige Algorithmen approximieren die Oberfläche anhand der Normalenvektoren der Punkte. Diese Methode führt zu glatten Oberflächen, bei denen die generierte Struktur möglicherweise nicht exakt durch alle Eingangspunkte verläuft, sondern eine optimierte, geglättete Fläche darstellt.
Andere Algorithmen verbinden immer exakt drei benachbarte Punkte zu einer Teilfläche, wodurch die erzeugte Oberfläche präzise durch die vorliegenden Punkte verläuft. Diese Methode kann zu einer detailgetreueren Repräsentation der Punktwolke führen, aber auch zu einer raueren Oberfläche, je nach Verteilung und Dichte der Punkte.
Im folgenden Umsetzungskapitel wird der Algorithmus ausgewählt, der am besten zu den spezifischen Anforderungen der Entwicklung passt und die gewünschten Eigenschaften der rekonstruierten Oberfläche gewährleistet. Die Wahl des Algorithmus hängt von der Zielsetzung der Oberflächenrekonstruktion ab, wie beispielsweise der gewünschten Glätte der Oberfläche, der Genauigkeit der Punktrepräsentation und der Anwendung,

für die das 3D-Modell verwendet werden soll. Wie bei der lokalen Registrierung zuvor wird an dieser Stelle auf ein Ablaufdiagramm verzichtet.

6 Umsetzung

In diesem Kapitel wird die Umsetzung der im vorherigen Kapitel beschriebenen Vorgehensweise detailliert erläutert. Zunächst erfolgt die Darstellung des Aufbaus und der Entwicklung der Sensoreinheit. Im Anschluss daran wird der Aufnahmebereich definiert, womit die Grundlage für die folgenden Punkte geschaffen ist. Danach wird die Entwicklung der Software erläutert, wobei die einzelnen Schritte und Komponenten in Übereinstimmung mit dem zuvor vorgestellten Ablaufdiagramm nach Funktionsgruppen geordnet beschrieben werden. Dieses strukturierte Vorgehen stellt sicher, dass der gesamte Entwicklungsprozess nachvollziehbar bleibt und die Integration von Hardware und Software effizient und kohärent erfolgt.
Die Basis der Softwareentwicklung bildet das GUI. Dieses wird zunächst grundlegend aufgesetzt und anschließend schrittweise mit jeder neuen Teilkomponente erweitert. Jede Teilkomponente wird dabei umfassend beschrieben, alternative Ansätze werden verglichen, die am besten geeignete Methode wird ausgewählt und anschließend direkt in das bestehende GUI integriert. Nach der Implementierung jeder Teilkomponente wird das aktualisierte GUI vorgestellt und das erzielte Ergebnis, das durch die jeweilige Teilkomponente erreicht wurde, dargestellt. Durch dieses iterative Vorgehen wird die GUI Schritt für Schritt um die notwendigen Komponenten erweitert, bis am Ende eine voll funktionsfähige und umfassende Benutzeroberfläche entwickelt wurde, die alle Anforderungen des Projekts erfüllt.

6.1 Sensoreinheit

Wie im Konzeptkapitel beschrieben, wird ein Griffstück konstruiert, an dem die Kameras befestigt werden, um diese um das zu erfassende Objekt zu führen. Der Aufbau wird dabei modular gestaltet. Das bedeutet, dass das Griffstück separat von den Kamerahalterungen konstruiert wird und die Komponenten später mit Schrauben verbunden

werden. Dieser modulare Ansatz ermöglicht es, verschiedene Kamerahalterungen auszuprobieren, die sich in der Ausrichtung der Kameras unterscheiden. Dadurch können sowohl Material als auch Zeit gespart werden, da Anpassungen und Optimierungen an der Kameraausrichtung vorgenommen werden können, ohne dass die gesamte Einheit neu gedruckt werden muss. Dieser flexible Aufbau erleichtert die iterative Entwicklung und ermöglicht es, die optimale Konfiguration für die spezifischen Anforderungen des Projekts zu finden.

6.1.1 Kamerahalterungen

In Abbildung 6.1 ist ein Teil der technischen Zeichnungen der D415-Kameras dargestellt. Daraus geht hervor, dass die Kameras auf der Rückseite über zwei M3-Gewinde verfügen, mit denen sie befestigt werden können. Zusätzlich besitzen diese Kameras ein Stativgewinde an der Unterseite.
Im Folgenden werden die M3-Gewinde zur Befestigung der Kamera an der Kamerahalterung verwendet. Schrauben mit einem M3-Gewinde sind im Gegensatz zu Stativschrauben leicht verfügbar. Darüber hinaus bleibt das Stativgewinde frei zugänglich, sodass es später verwendet werden kann, um die Sensoreinheit in einer festen Position zu fixieren. Dies ist besonders vorteilhaft bei Tests während der Softwareentwicklung.
Um sicherzustellen, dass die erzeugten Daten später intuitiv verarbeitet werden können, werden die beiden Kameras so montiert, dass ihre Koordinatensysteme, abgesehen von einer gewissen Rotation und Translation, gleich ausgerichtet sind. Das bedeutet, dass die Oberseiten der Kameras in die gleiche Richtung zeigen, sodass die Punktwolken der einen Kamera nicht jedes Mal um 180 Grad gedreht werden müssen. Dies erfordert, dass die untere Kamerahalterung über eine Aussparung verfügt, in der die Kabel, die zur Synchronisierung der beiden Kameras dienen, verlegt werden können.
In der ersten Iteration werden die Kamerahalterungen in Relation zum Griffstück nicht gekippt, sodass der Winkel zwischen den beiden Kameras ausschließlich durch die Geometrie des Griffstücks bestimmt wird. Diese Ausrichtung stellt den Basiswert dar, der in Zukunft durch Anpassungen des Winkels der Kamerahalterungen modifiziert werden kann. In Abbildung 6.4 sind die Modelle beider Kameraaufnahmen in ihrer relativen Ausrichtung dargestellt.

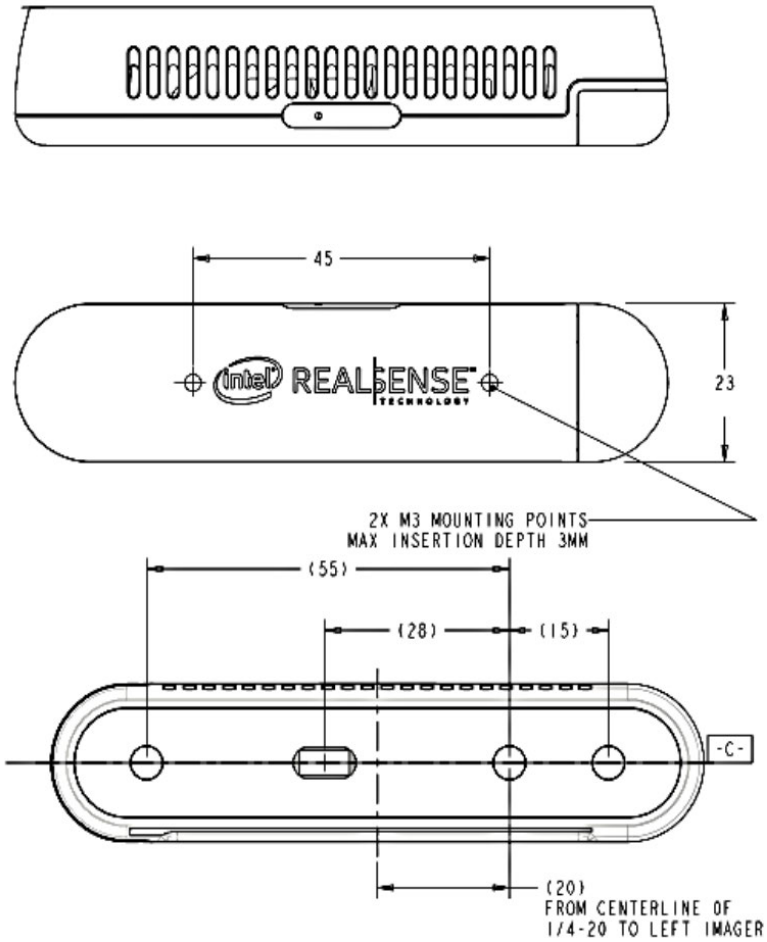


Abbildung 6.1: Auszug aus den Technischen Zeichnungen der D415 Kameras [20]

6.1.2 Griffstück

Das Griffstück wurde mit einem ergonomischen Fingerprofil konstruiert, das dem kommerzieller Produkte nachempfunden ist. Die Dimensionen wurden dabei nach Messungen der Hand des Erstellers dieser Arbeit angepasst, um eine bequeme und sichere Handhabung während der Nutzung zu gewährleisten. An den beiden Enden des Griffstücks befinden sich glatte Flächen, auf denen die Kameraaufnahmen aufliegen können. Zur Befestigung der Kameraaufnahmen wurden Einschmelzgewinde für M4-Schrauben verwendet, was eine stabile und wiederholbare Montage ermöglicht. In Abbildung 6.3 ist das 3D-Modell des Griffstücks dargestellt.

(a) Kameraaufnahme oben (b) Kameraaufnahme unten

Abbildung 6.2: 3D-Modelle der Kameraaufnahmen

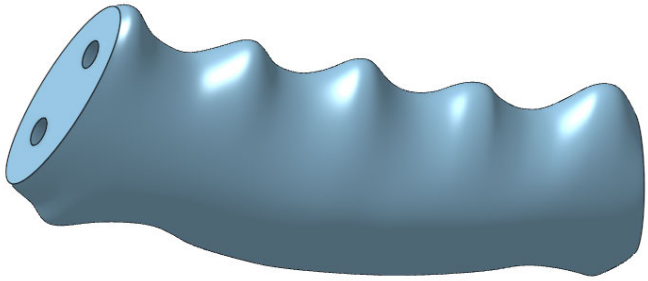
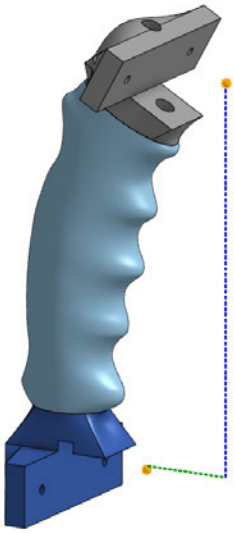


Abbildung 6.3: 3D-Modell des Griffstücks

6.1.3 Ergebnis

Aus der Geometrie des Griffstücks ergibt sich ein Winkel von 29 Grad zwischen den beiden Auflageflächen der Kameraaufnahmen. Mit den verbauten Kameraaufnahmen ergibt sich für die Ursprünge der Koordinatensysteme der Kameras, von der rechten Seite betrachtet, ein horizontaler Abstand von etwa 33 mm und ein vertikaler Abstand von etwa 148 mm. In Bezug auf die andere Achse gibt es keinen Versatz zwischen den Kameras.
Aufgrund dieser geometrischen Eigenschaften und des vom Kamerahersteller angegebenen vertikalen Sichtfelds befindet sich der Schnittpunkt der Sichtfelder in einer Entfernung von etwa 11 cm vom Zentrum der Sensoreinheit. Das bedeutet, dass Punkte, die sich mindestens 11 cm vom Zentrum der Sensoreinheit entfernt befinden, von beiden Kameras erfasst werden können.
Beim 3D-Druck wird der sogenannte „Infill“ verwendet, um die innere Struktur und Dichte eines Objekts zu definieren. Durch die Auswahl eines geringeren Infill-Werts kann das Gewicht des gedruckten Objekts erheblich reduziert werden, was insbesondere bei Anwendungen nützlich ist, die keine hohen Anforderungen an die Festigkeit stellen. So liegt das berechnete Materialgewicht des Griffstücks und der Kameraaufnahmen bei



(a) Zusammengefügtes Modell mit Koordinatenursprüngen

(b) 3D-gedruckte Sensoreinheit mit eingebauten Kameras und Synchronisierungsleitungen

Abbildung 6.4: Komplette Sensoreinheit

100 % Infill bei 153 Gramm, während es bei einer Infill-Dichte von 20 % lediglich 57 Gramm beträgt. Ein weiterer Vorteil des geringeren Infill-Werts ist die deutlich reduzierte Druckdauer, die sich bei 100 % Dichte mehr als vervierfacht. Die Materialkosten für alle Teile belaufen sich auf ungefähr 1,17 €, was das Drucken dieser Komponenten äußerst kosteneffizient macht [29].
In Abbildung 6.4a ist das vollständige 3D-Modell der Sensoreinheit dargestellt, inklusive der Koordinatenursprünge beider Kameras. In Abbildung 6.4b ist die gedruckte Sensoreinheit mit den eingebauten Kameras abgebildet. Die beiden roten Leitungen werden für die zeitliche Synchronisierung verwendet.

6.2 Aufnahmebereich

Mit der ausgewählten Kamera können Tiefenbilder unter verschiedenen Beleuchtungsbedingungen, einschließlich direkter Sonneneinstrahlung, aufgenommen werden. Die Genauigkeit der erfassten Tiefeninformationen hängt jedoch maßgeblich von der Texturierung des Aufnahmebereichs und des zu scannenden Objekts ab. Die Tiefeninformationen werden durch das Finden von Korrespondenzen in den Bildern der beiden Kameras ermittelt. Dies bedeutet, dass ein Punkt in beiden Bildern exakt identifiziert werden muss,

um eine genaue Tiefenmessung zu gewährleisten.
Die Genauigkeit der Tiefenbilder kann durch eine gezielte Variation der Texturierung verbessert werden. Eine optimierte Texturierung erhöht die Wahrscheinlichkeit, dass identische Punkte in beiden Bildern korrekt zugeordnet werden, was zu präziseren Tiefeninformationen führt. In [45] stellt der Hersteller der Kamera Rauschmuster zur Verfügung, die eine optimale Texturierung für die Tiefenmessung bieten. Ebenfalls wird darin beschrieben, dass die Verwendung eines Projektors zur Texturierung der aufgenommenen Szene die Aufnahmequalität erheblich verbessern kann. Während das Objekt in den meisten Fällen nur durch externe Texturierung strukturiert werden kann, lässt sich der Aufnahmebereich direkt so auswählen, dass er optimal geeignet ist. Für die folgenden Schritte, insbesondere bei der Vorverarbeitung, ist es von Vorteil, wenn auch der Aufnahmebereich möglichst gut texturiert ist, um eine exakte Identifikation zu ermöglichen.
Im Folgenden wird untersucht, welche Beschaffenheit der Aufnahmebereich aufweisen sollte, um optimale Ergebnisse bei der Datenerfassung zu erzielen. Dabei werden die folgenden Kombinationen von Texturierungs- und Beleuchtungsbedingungen verglichen.

1. Im ersten Versuch wird der Aufnahmebereichs mit weißem Papier ausgelegt, kombiniert mit einer Beleuchtung durch Deckenleuchten als auch durch natürliches Licht von Fenstern. Diese Konstellation bietet eine gleichmäßige, diffuse Beleuchtung, die jedoch in Kombination mit dem weißen Papier zu einer eher schwachen Texturierung führen kann. Dabei bildet dieser Fall den Zustand ab, mit dem auch das zu scannende Objekt aufgenommen wird, wenn dieses nicht mit einem Projektor bestrahlt wird.
   Da die Kamera bei der Tiefenbestimmung auf markante Bildmerkmale angewiesen ist, könnte die homogene weiße Oberfläche die Genauigkeit der Tiefenmessung beeinträchtigen. Insbesondere in Bereichen ohne klare Texturen könnte es schwierig sein, genaue Korrespondenzen zwischen den Bildern der beiden Kameras zu finden, was zu einer erhöhten Unsicherheit in den Tiefeninformationen führen könnte.

2. Der zweite Versuch besteht darin, den Aufnahmebereich mit weißem Papier auszulegen und zusätzlich ein Rauschmuster mit einem Beamer auf die Fläche zu projizieren. Diese Methode wird bei abgeschalteter Raumbeleuchtung und abgedunkelten Fenstern durchgeführt, um den Einfluss von externem Licht zu minimieren und die projizierten Muster klar und deutlich auf der weißen Oberfläche sichtbar zu machen.

Das projizierte Rauschmuster erzeugt eine künstliche Texturierung, die dazu beiträgt, markante Bildmerkmale im Aufnahmebereich zu erzeugen. Diese Merkmale erleichtern es der Kamera, genaue Korrespondenzen zwischen den beiden aufgenommenen Bildern zu finden, was die Präzision der Tiefenmessung erheblich verbessern kann.

3. Im letzten Versuch wird der Aufnahmebereich mit Ausdrucken des Rauschmusters ausgelegt, während die Aufnahme unter üblicher Raumbeleuchtung mit Deckenleuchten und natürlichem Licht durch Fenster erfolgt. Diese Methode kombiniert die Vorteile einer texturierten Oberfläche mit einer alltäglichen Beleuchtungssituation.
   Die Verwendung von ausgedruckten Rauschmustern erzeugt eine konsistente und gleichmäßige Texturierung, die es der Kamera erleichtert, markante Bildmerkmale zu identifizieren und genaue Korrespondenzen zwischen den beiden aufgenommenen Bildern zu finden. Durch die Kombination dieser strukturierten Oberfläche mit der natürlichen und künstlichen Raumbeleuchtung wird eine Umgebung geschaffen, die typische Einsatzbedingungen simuliert, während gleichzeitig die Genauigkeit der Tiefenmessung durch die verbesserte Texturierung maximiert wird.

6.2.1 Bewertungsmetrik und Versuchsaufbau

Die Konzepte werden durch eine statische Datenaufnahme getestet. Dabei wird eine Kamera auf einem Stativ positioniert, sodass der gesamte Aufnahmebereich vollständig im Bild erfasst wird. Zur Analyse der aufgenommenen Daten wird die Open3D-Funktion *segment_plane()* verwendet. Diese Funktion nutzt den RANSAC-Algorithmus, um Ebenen in einer Punktwolke zu segmentieren. Dabei werden in N Iterationen jeweils drei zufällige Punkte aus der Punktwolke ausgewählt, um eine Ebene zu rekonstruieren. Anschließend wird für jeden Punkt in der Punktwolke überprüft, ob er Teil dieser Ebene ist. Hierfür wird ein Threshold in Metern definiert, der den maximalen Abstand eines Punktes zur rekonstruierten Ebene angibt, bis zu dem dieser Punkt als Teil der Ebene klassifiziert wird.
In der vorliegenden Untersuchung werden 100.000 Iterationen durchgeführt, wobei in jeder Iteration eine andere Kombination von Punkten überprüft wird, um die bestmögliche Ebene zu identifizieren. Der Threshold wird auf 0,001 Meter (1 Millimeter) festgelegt. Das bedeutet, dass ein Punkt nur dann als Teil der rekonstruierten Ebene anerkannt wird, wenn sein Abstand zur Ebene weniger als einen Millimeter beträgt.

Die Leistung der Projektoren ist über die SDK von 0 bis 360 mW einstellbar. Die Datenerfassung erfolgt unter Nutzung der vollen Laserleistung beider Kameras, wobei allerdings nur das Bild einer Kamera für die Analyse verarbeitet wird.
Die Bewertung des Versuchs erfolgt anhand der Anzahl der Punkte in der Punktwolke, die als Teil der rekonstruierten Ebene klassifiziert werden. Diese Metrik ermöglicht es, das Verfahren zu identifizieren, das den Aufnahmebereich am genauesten als Ebene erkennt. Durch diesen Vergleich lässt sich bestimmen, welche der getesteten Texturierungs- und Beleuchtungskonzepte die präziseste Erfassung der Fläche ermöglicht.

6.2.2 Ergebnis

In Abbildung 6.5 ist exemplarisch für alle durchgeführten Versuche die rekonstruierte Fläche für die dritte Kombination dargestellt. Die in Rot hervorgehobenen Punkte zeigen diejenigen an, die erfolgreich der Fläche zugeordnet werden konnten.
Von den getesteten Kombinationen erzielte die zweite die schlechtesten Ergebnisse. Bei dieser Aufnahme konnten lediglich etwa 40 % der Punkte der rekonstruierten Fläche zugeordnet werden. Dieses Ergebnis könnte darauf zurückzuführen sein, dass die Beleuchtung ausschließlich durch den Beamer erfolgte, wodurch der Aufnahmebereich möglicherweise nicht ausreichend ausgeleuchtet wurde. Insbesondere bei Tageslicht war das projizierte Bild aufgrund der begrenzten Leuchtkraft des Beamers kaum sichtbar, was die Genauigkeit der Tiefenerfassung beeinträchtigte.
Die erste Kombination führte zu einer leicht verbesserten Rekonstruktion, bei der etwa 45 % der Punkte der Fläche zugeordnet werden konnten. Trotz der geringen Texturierung im sichtbaren Bereich konnte eine Verbesserung gegenüber der abgedunkelten Aufnahme erzielt werden. Dies deutet darauf hin, dass selbst nicht exakt kontrollierte Lichtbedingungen eine positive Wirkung auf die Erfassung der Tiefendaten haben können.
Das beste Ergebnis wurde mit der dritten Kombination erzielt, bei der etwa 56 % der Punkte erfolgreich einer gemeinsamen Fläche zugeordnet werden konnten. Entscheidend für dieses Ergebnis dürfte die optimale Texturierung im sichtbaren Spektrum gewesen sein, die durch die Beleuchtung mit einer Kombination aus künstlichem und natürlichem Licht erreicht wurde.
Aufgrund dieser Ergebnisse wird für die weitere Entwicklung der Aufnahmebereich mit dem Rauschmuster ausgelegt. Um diesen Vorgang sowohl für den Anwender als auch für den Entwickler zu optimieren, wurden vier Hartschaumplatten mit den Maßen 50

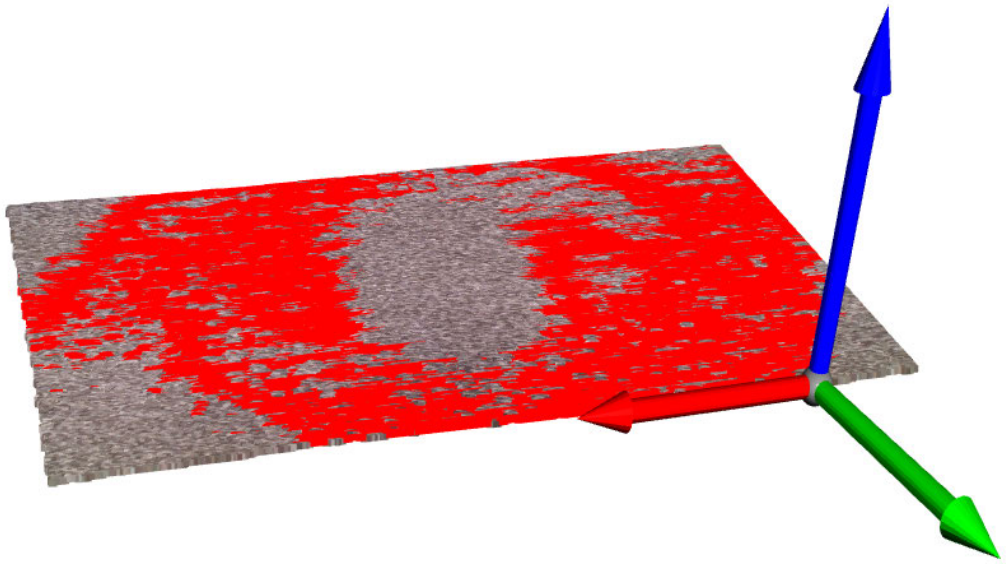


Abbildung 6.5: Rekonstruierte Fläche (Rot eingefärbt: Punkte, die als Teil der am besten passenden Fläche klassifiziert wurden.

x 50 cm und einer Dicke von 5 mm mit dem Rauschmuster bedruckt. Die Kosten für diese Platten betrugen 55,36 €, mit denen das Projektbudget belastet wurde.

6.3 Software Übersicht

In den folgenden Abschnitten wird die Softwareentwicklung erläutert. Zunächst werden die einzelnen Komponenten separat beschrieben. In Abschnitt 6.14 folgt anschließend ein Überblick über den Gesamtzusammenhang aller Teilkomponenten sowie eine Beschreibung ihrer Interaktionen untereinander.

6.4 User Interface Grundlage

Mit dem Qt Designer wird zunächst ein Objekt der Klasse *QMainWindow* erstellt. Dieses Objekt stellt das Hauptfenster dar, in dem im Laufe der Entwicklung weitere Objekte, im Kontext der GUI-Erstellung als Widgets bezeichnet, hinzugefügt werden. Zu Beginn enthält die Klasse lediglich das Hauptfenster, das noch keine interaktiven

Elemente umfasst.
Der Qt Designer erstellt Dateien mit der Endung .ui, die das Layout und die Struktur der Benutzeroberfläche definieren. Diese .ui-Dateien können entweder direkt im Hauptprogramm geladen oder mithilfe des Befehls *pyuic6* in Python-Skripte umgewandelt werden. Diese Skripte enthalten die durch den Qt Designer erstellten Klassen, die dann im Hauptprogramm instanziiert und genutzt werden können.
Im Hauptprogramm können die sogenannten Signale und Slots der Klassenobjekte genutzt werden, um eine ereignisgesteuerte Steuerung der Benutzeroberfläche zu ermöglichen. Signale sind spezielle Ereignisse, die beispielsweise durch Benutzerinteraktionen wie das Klicken eines Buttons ausgelöst werden. Slots sind Funktionen, die auf diese Signale reagieren und entsprechende Aktionen ausführen. Durch die Verbindung von Signalen und Slots wird ein ereignisgesteuerter Ablauf entwickelt, der die Interaktivität und Funktionalität der Benutzeroberfläche definiert.

6.5 Prozess konfigurieren

Dieser erste Schritt bei der Benutzung der Software umfasst die Konfiguration des Aufnahmeprozesses, bevor die ersten Daten erfasst werden. Um eine grundlegende Funktionalität sicherzustellen, auf der die nachfolgenden Schritte aufbauen können, werden an dieser Stelle einige wesentliche Einstellungen in die Benutzeroberfläche integriert. Diese frühe Implementierung von Einstellmöglichkeiten ermöglicht es, die Software in einem minimal funktionsfähigen Zustand zu nutzen, während die restlichen Komponenten schrittweise entwickelt und hinzugefügt werden. Die folgenden Interaktionsmöglichkeiten werden in die GUI implementiert:

1. Verbindung zu Kameras herstellen: Der Benutzer kann mit einem Taster die Verbindung zu den Kameras herstellen.

2. Anzahl der Aufnahmen: Der Benutzer kann festlegen, wie viele Einzelpunktwolken während des Aufnahmeprozesses erstellt werden sollen. Diese Einstellung ist entscheidend, um sicherzustellen, dass genügend Daten gesammelt werden, um das Objekt vollständig zu rekonstruieren.

3. Auflösung und Bildrate: Es werden Optionen zur Auswahl der Auflösung und Bildrate der Kameras bereitgestellt.

4. Kalibrierung erwünscht oder nicht: Der Benutzer kann durch einen Schalter festlegen, ob vor der Datenaufnahme eine Kalibrierung der Sensoreinheit durchgeführt werden soll oder ob die aktuellen Kalibrierparameter genutzt werden sollen.

5. Anzahl der Kalibrieraufnahmen: Der Benutzer kann festlegen, wie viele Aufnahmen zur Kalibrierung aufgenommen werden sollen.

6. Einstellungsdatei auswählen: Der Benutzer kann eine Datei auswählen, in der die Einstellungen definiert sind, mit denen die Kamera genutzt werden soll.

7. Speicherort der Daten: Die UI ermöglicht es dem Benutzer, den Speicherort für das rekonstruierte Objekt zu definieren.

8. Livebild: Es wird ein Fenster zur Ausgabe des Livebildes beider Kameras hinzugefügt. Dieses Livebild ermöglicht es dem Benutzer, die aktuelle Ansicht und den Status der Kameras in Echtzeit zu überwachen, was insbesondere bei der Ausrichtung hilfreich ist.

9. Prozess starten: Mit dieser Schaltfläche wird der Aufnahmeprozess gestartet

10. Beenden: Wenn der Benutzer die Schaltfläche betätigt, wird das Programm beendet und die Verbindung zu den Kameras getrennt.

In Abbildung 6.6 ist die Basisversion der Benutzeroberfläche mit den zuvor beschriebenen Interaktionsmöglichkeiten dargestellt. In dieser ersten Version ist zunächst ausschließlich die Funktion zum Schließen des Fensters mit dem Taster *Beenden* implementiert. Diese Version der GUI bildet die Grundlage für die Integration der weiteren Funktionen, die im Laufe der Entwicklung hinzugefügt werden.

6.6 Einrichtung der Hardware (CameraManager)

Die Software für dieses Projekt wird, wo es sinnvoll ist, objektorientiert und mit klaren Verantwortlichkeiten entwickelt. Für die Vorgänge, bei deren Ausführung ein zugriff auf die Kameras notwendig ist, wird die Klasse *CameraManager* erstellt, die diese Methoden implementiert. Im Folgenden werden zunächst die Methoden beschrieben. Zur Visualisierung ist in Abbildung 6.7 das entsprechende Klassendiagramm dargestellt. Abschließend wird die Implementierung der Benutzeroberfläche beschrieben.

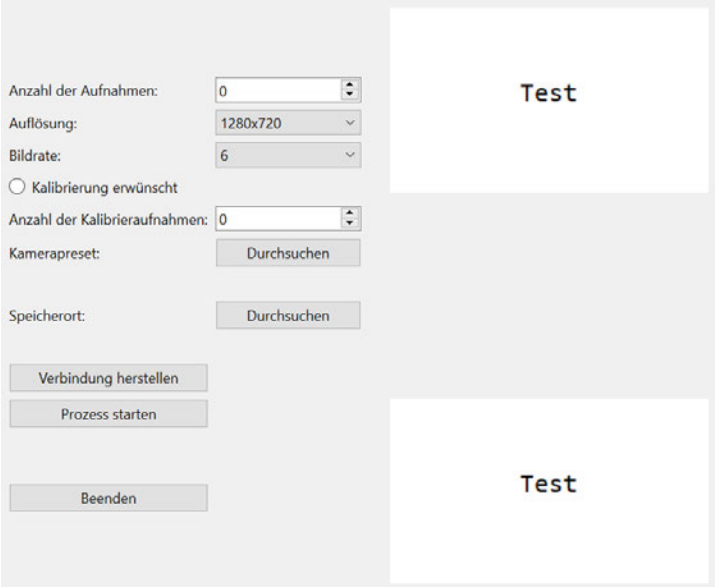


Abbildung 6.6: Basisversion des UI

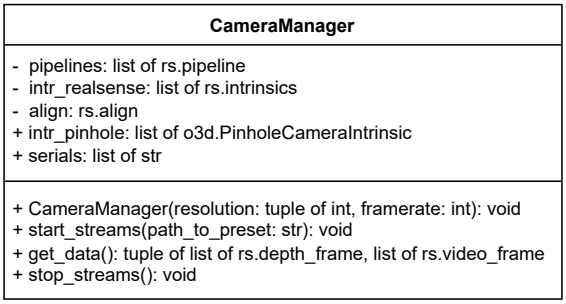


Abbildung 6.7: Klassendiagramm der Klasse *CameraManager*

6.6.1 Konstruktor

Beim Erstellen einer Instanz der Klasse wird der Konstruktor aufgerufen, welcher die vom Benutzer spezifizierten Einstellungen für Auflösung und Bildrate entgegennimmt. Der Konstruktor prüft zunächst, ob zwei Kameras verbunden sind. Sind zwei Kameras angeschlossen, wird die erste als Master und die zweite als Slave konfiguriert. Diese Zuweisung ermöglicht die Synchronisation der Kameras: Sobald die Master-Kamera ein Bild aufnimmt, wird gleichzeitig der Aufnahmevorgang der Slave-Kamera initiiert. Nach Abschluss dieser Konfiguration wird für jede Kamera eine Pipeline eingerichtet, womit die Instanziierung der Klasse vollendet ist.

6.6.2 Starten der Streams

Mit der Methode *start_ streams* wird der Datenstrom über die beiden Pipelines gestartet, sodass fortan Daten abgerufen werden können. In diesem Schritt erfolgt auch die Einstellung bzw. Übernahme der Kamerakonfiguration.
Die RealSense-Kameras bieten eine breite Palette an einstellbaren Parametern, die es dem Benutzer ermöglichen, die Kameras optimal an den jeweiligen Anwendungszweck anzupassen. Der Hersteller stellt zudem verschiedene Presets zur Verfügung, die es ermöglichen, die Kameras bequem und effizient mit einem vordefinierten Parametersatz zu konfigurieren. Für die weitere Entwicklung wurde entschieden, dass es nicht zielführend ist, diese Parametrierung direkt in der entwickelten Software zu implementieren. Stattdessen erfolgt die Einstellung der Parameter über die vom Hersteller bereitgestellte Software.
Mit der eigenständigen Software *RealSense Viewer* können die RealSense-Kameras verbunden und konfiguriert werden. Sobald mindestens eine Kamera erkannt und verbunden ist, kann die Parametrierung durchgeführt werden. Der erste Schritt besteht darin, ein Preset auszuwählen. Abbildung 6.8 zeigt diesen Vorgang. Für die vorliegende Anwendung ist das Preset *High Accuracy* besonders empfehlenswert. Dieses Preset fokussiert sich auf die Erzeugung von Punkten, deren Tiefenwerte mit hoher Zuverlässigkeit ermittelt wurden. Das Ergebnis ist eine weniger dichte, aber dafür präzisere Punktwolke. Auf Basis des gewählten Presets können anschließend Parameter angepasst werden. Dabei wird grundsätzlich zwischen Parametern, die die Stereokameras betreffen, und solchen, die die RGB-Kamera betreffen, unterschieden. Im Folgenden werden die wesentlichen Parameter beschrieben, die zur Konfiguration der Stereokameras relevant sind.

1. Belichtungszeit (Exposure): Diese Einstellung legt die Dauer fest, für die die Kameras Licht einfangen, um ein Bild zu erzeugen. Zur Vereinfachung kann eine automatische Belichtungszeit gewählt werden, die sich an die Umgebungsbedingungen anpasst. In bestimmten Situationen, insbesondere wenn eine höhere Genauigkeit der Tiefendaten erforderlich ist, kann es jedoch vorteilhaft sein, die Belichtungszeit manuell anzupassen. Eine spezifisch eingestellte Belichtungszeit ermöglicht es, die Kamera optimal auf die jeweilige Szenerie abzustimmen, was die Qualität und Präzision der erfassten Tiefendaten verbessern kann.

2. Laserleistung (Laser Power): Diese Einstellung reguliert die Intensität des Infrarotprojektors, der zur Texturierung des Aufnahmebereichs verwendet wird. Im

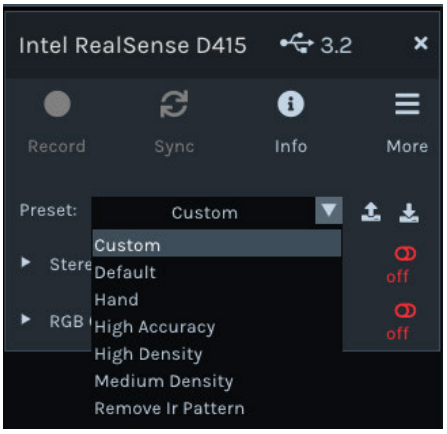


Abbildung 6.8: Auswahl des Preset im RealSense Viewer

Allgemeinen sollte die Laserleistung möglichst hoch eingestellt werden, um die Qualität der Tiefendaten zu maximieren. Es ist jedoch zu berücksichtigen, dass eine hohe Leistung, je nach Szenerie und dem Abstand der Kamera zum Objekt, zu einer unerwünschten Streuung des Infrarotlichts führen kann. In [16] wird beschrieben, dass diese Streuung durch den sogenannten Speckle-Effekt verursacht werden könnte. Darüber hinaus wird in der Quelle erläutert, unter welchen Bedingungen dieser Effekt auftritt und wie er die Genauigkeit der Tiefendaten beeinflussen kann. Daher ist es wichtig, die Laserleistung an die spezifischen Bedingungen der Szene anzupassen, um eine optimale Balance zwischen Beleuchtung und Streuung zu erreichen.

3. Tiefenauflösung (Depth Units): Mit diesem Parameter wird die Auflösung der Tiefenmessung in Metern eingestellt. Der Standardwert entspricht einer Auflösung von einem Millimeter, was für viele Anwendungen ausreichend ist. Für den vorliegenden Anwendungsfall sollte jedoch eine höhere Präzision angestrebt werden, weshalb die Tiefenauflösung auf 100 Mikrometer (0,0001 Meter) eingestellt werden sollte. Dies ermöglicht eine feinere Erfassung von Tiefeninformationen. Es ist jedoch zu beachten, dass durch die Verwendung eines 16-Bit-Wertes zur Darstellung der Tiefeninformationen der maximale darstellbare Tiefenbereich auf etwa 6,5 Meter reduziert wird. Diese Anpassung stellt sicher, dass innerhalb des relevanten Messbereichs eine hohe Genauigkeit gewährleistet ist, während tiefere Bereiche möglicherweise nicht mehr vollständig erfasst werden können.

4. Disparitätsverschiebung (Disparity Shift): Dieser Parameter ermöglicht es, den minimalen Aufnahmeabstand der Kamera zu verringern, allerdings auf Kosten des maximal erfassbaren Abstands. Bei Standardeinstellungen beträgt der minimale Aufnahmeabstand etwa 45 cm. Wird der Disparitätsverschiebungswert jedoch auf 50 gesetzt, verringert sich der Aufnahmeabstand auf einen Bereich zwischen 30 und 110 cm. Da die Tiefendaten umso präziser sind, je näher die Kamera am Objekt positioniert ist, kann durch die Anpassung dieses Parameters eine höhere Genauigkeit der Aufnahmen erreicht werden.

Weitere Optimierungsmöglichkeiten der Parameter sind in [45] beschrieben. Es wird empfohlen, diesen Artikel bei der Parametrierung heranzuziehen, um fundierte Entscheidungen zu treffen. Zudem ist es ratsam, die Auswirkungen der verschiedenen Einstellungen in einem Livebild zu beobachten, um die optimalen Parameter für die jeweilige Szenerie zu ermitteln. Der *RealSense Viewer* bietet die Möglichkeit, ein Livebild der erfassten Daten anzuzeigen, wobei die Tiefendaten mit den aufgenommenen Farbwerten überlagert werden können. Auf diese Weise können Benutzer direkt sehen, wie sich die Anpassungen auf die Bildqualität auswirken. Sobald die Parametrierung den gewünschten Anforderungen entspricht, kann das optimierte Preset als JSON-Datei exportiert werden.

Beim Ausführen der Methode *start_streams* wird die im Benutzerinterface angegebene Preset-Datei geladen und auf die Kameras angewendet. Nach dieser Konfiguration sind die Kameras bereit, um Daten aufzunehmen.

6.6.3 Datenaufnahme

Die Methode *get_data* wird verwendet, um von beiden Kameras jeweils ein Tiefenbild (Depth Frame) und ein Farbbild (Color Frame) aufzunehmen. Hierzu wird die Funktion *wait_for_frames* des RealSense SDK auf jede Pipeline angewendet, wodurch ein Frameset zurückgegeben wird, das sowohl den Depth Frame als auch den Color Frame enthält. Da diese Bilder außerhalb der RealSense-Funktionen nicht direkt verarbeitet werden können, müssen sie an einer späteren Stelle aus dem Frameset extrahiert werden. Die Frames werden an dieser Stelle gespeichert, da sie für andere Funktionen, die das RealSense SDK nutzen, weiterhin relevant sind. Dazu werden beide Depthframes und beide Colorframes jeweils in eine Liste eingefügt. Diese Listen werden dann an die

aufrufende Stelle zurückgegeben, sodass die aufgenommenen Daten weiterverarbeitet werden können.

6.6.4 Datenübertragung beenden

Mit der Methode *stop_streams* wird der Datenstrom über beide Pipelines beendet, sofern dieser zuvor gestartet wurde. Dies stellt sicher, dass alle laufenden Prozesse ordnungsgemäß abgeschlossen werden und die Ressourcen freigegeben werden, wodurch ein reibungsloses Beenden des Streaming-Vorgangs gewährleistet wird.

6.6.5 Änderungen im UI

Alle implementierten Methoden wurden, da wo es notwendig war, in die Benutzeroberfläche (UI) integriert. Im Rahmen dieser Integration wurden folgende Änderungen und Ergänzungen an der Benutzeroberfläche vorgenommen:

1. Die Auswahlfelder für Auflösung und Bildrate wurden mit den verfügbaren, einstellbaren Werten gefüllt, sodass der Benutzer diese Parameter direkt in der Benutzeroberfläche anpassen kann.

2. Die Schaltfläche *Durchsuchen* wurde mit einer Methode verknüpft, die beim Anklicken ausgeführt wird. Beim Auslösen dieser Methode öffnet sich ein neues Fenster mit einem Dateiauswahldialog. In diesem Dialog kann der Benutzer die gewünschte Preset-Datei auswählen, woraufhin der Pfad zur Datei gespeichert wird.

3. Beim Klicken auf die Schaltfläche *Verbindung herstellen* wird ein Objekt der Klasse *CameraManager* mit den aktuell ausgewählten Werten für Auflösung und Bildrate instanziiert. Im Falle eines erfolgreichen Verbindungsaufbaus öffnet sich ein Fenster, das die erfolgreiche Ausführung bestätigt.

4. Durch Klicken auf die Schaltfläche *Prozess starten* werden die Streams gestartet, vorausgesetzt, es wurde zuvor ein Preset ausgewählt. Diese Bedingung stellt sicher, dass die Kameras mit den gewünschten Einstellungen arbeiten, bevor der Streaming-Prozess beginnt.

5. Durch das Betätigen der Schaltfläche *Beenden* werden die Streams gestoppt, und gleichzeitig werden, wie zuvor, alle geöffneten Fenster geschlossen.

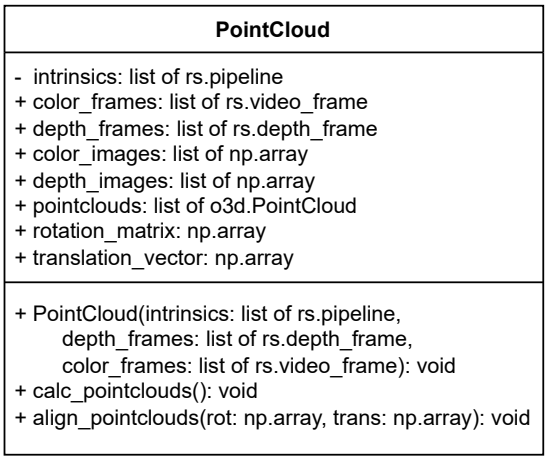


Abbildung 6.9: Klassendiagramm der Klasse *Pointcloud*

6.7 Punktwolkengenerierung: PointCloud

Um aus den Rohdaten Punktwolken zu generieren, wird die Klasse *PointCloud* implementiert. Diese Klasse übernimmt die Aufgabe, die erfassten Tiefen- und Farbdaten in eine 3D-Punktwolke zu konvertieren, wodurch das Objekt im dreidimensionalen Raum dargestellt werden kann. Neben der Erzeugung der Punktwolken enthält die Klasse eine Methode, die es ermöglicht, gleichzeitig aufgenommene Punktwolken, die von verschiedenen Kameras stammen, präzise zueinander auszurichten. In Abbildung 6.9 ist das entsprechende Klassendiagramm dargestellt. Eine Änderung im UI ist nicht erfolgt.
Für dieses und die folgenden Abschnitte, in denen die aufgenommenen Punktwolken verarbeitet werden, werden die Änderungen an den aufgenommenen Punktwolken am Ende der jeweiligen Implementierung dargestellt. Als Aufnahmeobjekt dient dabei zunächst ein mit Kreppband beklebter Karton. Dieser hat den Vorteil, dass dessen einfache Geometrie einfach aufzunehmen und Änderungen daran gut darstellbar sind. Durch das Kreppband ist die Oberfläche für den Aufnahmeprozess besser geeignet, als es der glänzende Aufdruck war. Die Oberfläche ist von sich aus leicht Strukturiert und die Infrarotmuster werden aufgrund der matten Texturierung besser reflektiert.

6.7.1 Konstruktor

Bei der Instanziierung der Klasse werden die Depth- und Colorframes der jeweiligen Aufnahme übergeben. Zusätzlich werden die intrinsischen Kameraparameter bereitgestellt, die zur Erzeugung der Daten verwendet wurden. Falls bereits eine Kalibrierung der Kameras erfolgt ist, können die entsprechenden Transformationsanweisungen ebenfalls übergeben werden.
Aus den Depth- und Colorframes werden die Farb- und Tiefenbilder extrahiert und als Attribute der Klasse gespeichert. Diese Bilder dienen als Grundlage für die Generierung der Punktwolken.

6.7.2 Punktwolkenerstellung

Mit der Methode *calc_pointcloud* werden aus den zuvor erzeugten Bildern Punktwolken erstellt. Zunächst wird ein RGBD-Bild generiert, das aus den drei Farbkanälen (RGB) sowie einem zusätzlichen Tiefenkanal (D=Depth) besteht. Aus diesem RGBD-Bild wird anschließend mit der Open3D-Methode *create_from_rgbd_image* eine Punktwolke generiert. Dazu werden die intrinsischen Parameter der Kamera verwendet, mit denen das Farb- und Tiefenbild aufgenommen wurde. Diese Parameter sind notwendig, um die 2D-Pixel des Bildes zurück in 3D-Koordinaten zu transformieren.
Die erzeugten Punktwolken sind vom Typ *open3d.PointCloud*, auf den eine Vielzahl von Open3D-Methoden angewendet werden kann. So lassen sich beispielsweise die Koordinaten aller Punkte sowie die zugehörigen Farbwerte extrahieren und in einem Array speichern. Dies ermöglicht eine flexible Weiterverarbeitung der Punktwolkendaten, sei es durch eigene Methoden oder durch den Einsatz weiterer Open3D-Funktionen.

6.7.3 Punktwolkenausrichtung

Wenn durch eine vorherige Kalibrierung gültige Transformationsanweisungen in Form einer Rotationsmatrix und eines Translationsvektors vorliegen, können die Punktwolken einer Aufnahme zueinander ausgerichtet werden. Die Methode *align_pointclouds* bietet diese Funktionalität. Dabei werden die Open3D-Funktionen *rotate* und *translate* auf die Punktwolke angewendet, die an die andere Punktwolke angepasst werden soll.

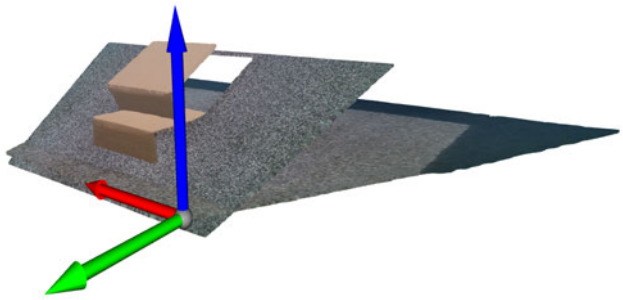


Abbildung 6.10: Unverarbeitete Punktwolken beider Kameras in ursprünglicher Ausrichtung

Durch die Rotation und Translation wird die Ausrichtung der beiden Punktwolken zueinander korrigiert. Im Anschluss wird eine neue Punktwolke erstellt, die die beiden Teilpunktwolken zu einer zusammengeführten Punktwolke kombiniert.

6.7.4 Ergebnis

In Abbildung 6.10 sind zwei bei einem Durchgang aufgenommene Punktwolken dargestellt. Die verschiedenen Punktwolken sind gut auszumachen, da diese so dargestellt sind, wie diese von den beiden Kameras aufgenommen werden. Dadurch sind diese rotiert und verschoben zueinander.

6.8 Kalibrierung der Sensoreinheit und Methodik zur Vorregistrierung

Dem Aktivitätsdiagramm in Abbildung 5.1 folgend wird nun die Kalibrierung der Sensoreinheit durchgeführt. Ziel dieser Kalibrierung ist es, Transformationsanweisungen zu ermitteln, die es ermöglichen, die von den beiden Kameras gleichzeitig aufgenommenen Punktwolken zueinander auszurichten. Da sich die relative Ausrichtung der Kameras nicht ändert, solange sie fest in der Sensoreinheit verbleiben, ist der Kalibrierungsprozess nur nach Entfernung oder Neujustierung der Kameras erforderlich, beispielsweise nach einem Wechsel der Kamerahalterungen.
Neben der Kamerakalibrierung bietet der beschriebene Prozess die Möglichkeit, die globale Vorregistrierung durchzuführen. Dieses Kapitel definiert die Methodik für diesen Prozess, während die Implementierung in einem späteren Kapitel behandelt wird.

Es werden drei Methoden zur Kalibrierung und Ausrichtung der Punktwolken miteinander verglichen. Alle basieren darauf, dass mit beiden Kameras jeweils ein Bild aufgenommen wird, in dem gleiche Schlüsselpunkte ermittelt werden. Die Unterschiede liegen in der Identifikation der Schlüsselpunkte und der Berechnung der optimalen Transformationsanweisung.
Eine Möglichkeit ist es, die Transformation direkt aus den zweidimensionalen Bildpunkten zu generieren, was die Kenntnis der intrinsischen Parameter erfordert. Diese Parameter beschreiben die Umwandlung von realen 3D-Koordinaten in 2D-Bildkoordinaten, sprich die Transformationsanweisung, mit der 3D-Punkte von der Kamera aufgenommen und in 2D-Bildpunkte transformiert werden. Außerdem müssen dafür die geometrischen Beziehungen der aufgenommenen Schlüsselpunkte zueinander im reellen Koordinatensystem bekannt sein. Für die Praxis bedeutet dies, dass Kalibriermuster verwendet werden müssen, bei denen die Positionen der Schlüsselpunkte untereinander definiert sind. Das ist zum Beispiel der Fall bei Kalibriertafeln. Diese, auch Objektpunkte genannten Informationen, müssen vor der Datenaufnahme und Berechnung der Transformation bekannt sein. Der Vorteil dieser Methode ist, dass die Punkte ausschließlich im Zweidimensionalen ermittelt werden müssen, was eine gute Genauigkeit erwarten lässt. Ein Nachteil ist, dass die Kalibriermuster im Aufnahmebereich nicht verteilt werden können, da dann die definierte Lage der Muster zueinander nicht mehr gegeben ist.
Alternativ können die Tiefenwerte der Schlüsselpunkte extrahiert und die daraus resultierenden 3D-Koordinaten verwendet werden, um die Transformation zu bestimmen.
Die optimale Transformationsanweisung wird dabei mit Hilfe der Kovarianzmatrix und der Singularwertzerlegung ermittelt. Der Nachteil dieser Methodik ist, dass die Genauigkeit abhängig ist von den ermittelten Tiefenwerten. Da diese durch das Tiefenrauschen beeinflusst werden, kann sich das auf die Genauigkeit der Transformationsanweisung auswirken. Durch das Aufnehmen von einer Vielzahl an Datenpunkten und dadurch, dass die Singularwertzerlegung robust gegenüber numerischen Ungenauigkeiten ist, kann diese Ungenauigkeit bis zu einem gewissen Maß ausgeglichen werden.
Zunächst werden die Bewertungsmetrik und der Versuchsaufbau beschrieben, gefolgt von einer detaillierten Beschreibung der Methoden. Abschließend werden die Ergebnisse verglichen und eine Entscheidung für die geeignetste Methodik getroffen.

6.8.1 Versuchsaufbau und Bewertungsmetrik

Für die nachfolgenden Untersuchungen werden 10 Sätze von jeweils zwei Bildern gleichzeitig mit beiden Kameras aufgenommen. Diese dienen zur Berechnung der Transformationsanweisung, die anschließend auf eine Teilpunktwolke angewendet wird. Die Bilder werden in einem statischen Szenario aufgenommen: Die Sensoreinheit wird auf einem Stativ befestigt und auf die Szene ausgerichtet, bevor das Stativ für jede neue Aufnahme in eine andere Position bewegt wird. Um die Vergleichbarkeit der Methoden zu gewährleisten, erfolgen alle Aufnahmen aus derselben Position, wobei der Aufnahmebereich den Anforderungen der jeweiligen Methodik angepasst wird.
Die Bewertung erfolgt visuell durch Betrachtung der zusammengefügten Punktwolken sowie quantitativ durch die Open3D-Funktion *evaluate_registration()*. Diese Funktion bewertet die Qualität der Registrierung, indem die Überlappung der Punktwolken analysiert wird. Die Funktion liefert einen Fitnesswert, der das Verhältnis der überlappenden Punkte zur Gesamtzahl angibt, sowie den RMS-Fehler, der den durchschnittlichen Abstand zwischen korrespondierenden Punkten beschreibt. Ein hoher Fitnesswert und ein niedriger RMS-Fehler sind wünschenswert.
Zusätzlich wird die Performance der Algorithmen untersucht. Die Zeit für die Berechnung der Punkte und der Transformationsanweisung wird ermittelt. Während die Ausführungszeit bei der Kalibrierung nicht kritisch ist, spielt sie bei der globalen Registrierung eine größere Rolle, da sie die Dauer des Gesamtprozesses beeinflusst.

6.8.2 Methode 1: Feature Matching mit ORB

Der ORB-Algorithmus wird eingesetzt, um Merkmale in zweidimensionalen Bildern zu erkennen und zu beschreiben. Er kombiniert den FAST-Algorithmus [76] zur Schlüsselpunkterkennung mit dem BRIEF-Deskriptor [15] zur Merkmalsbeschreibung. ORB wurde von OpenCV entwickelt und stellt eine effiziente Alternative zu den etablierten, aber rechenintensiveren Verfahren SIFT [58] und SURF [89] dar.
Die Schlüsselpunkte werden im ORB-Algorithmus mithilfe des FAST-Algorithmus (Features from Accelerated Segment Test) detektiert. Dabei wird jedes Pixel eines Bildes als potenzieller Schlüsselkandidat betrachtet. Um ein Pixel als Schlüsselpunkt zu identifizieren, wird ein Kreis von 16 Pixeln um das untersuchte Pixel mit Intensität I_p (siehe Abbildung 6.11) analysiert. Ein Pixel wird als Schlüsselpunkt klassifiziert, wenn mindestens n (typischerweise 12) aufeinanderfolgende Pixel in diesem Kreis existieren, deren

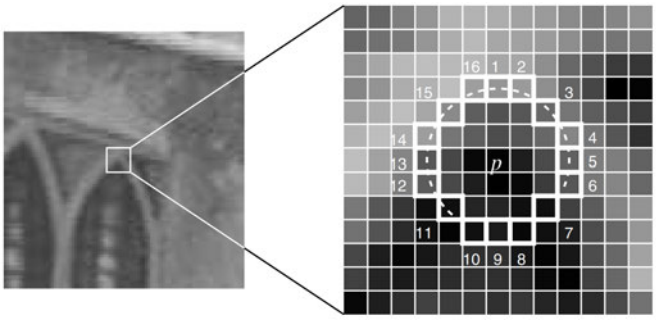


Abbildung 6.11: Kreis von 16 Pixeln um das betrachtete Pixel herum [76]

Intensitäten alle größer als $I_p + t$ oder alle kleiner als $I_p - t$ sind, wobei t ein Schwellwert ist, der auf das betrachtete Bild angepasst werden kann. Diese Vorgehensweise führt dazu, dass Ecken und Kanten detektiert werden.
Um die gefundenen Punkte zu beschreiben, wird der BRIEF-Deskriptor (Binary Robust Independent Elementary Features) verwendet. Dieser erstellt für jeden Schlüsselpunkt einen binären Merkmalsvektor, indem paarweise Intensitätsvergleiche innerhalb eines kleinen Bildbereichs um den Schlüsselpunkt durchgeführt werden. In ORB werden 32-Bit lange Merkmalsvektoren generiert. Da der ursprüngliche BRIEF-Deskriptor nicht rotationsinvariant ist, integriert ORB eine rotationsangepasste Version.

Anwendung

Da die gefundenen Schlüsselpunkte zufälliger Natur sind, können im Voraus keine festgelegten Objektpunkte ermittelt werden, die mit den gefundenen Bildpunkten übereinstimmen. Aus diesem Grund werden die Koordinaten direkt aus dem Farbbild gewonnen. Um gute Bedingungen für die Erkennung von Schlüsselpunkten zu schaffen, wird der Aufnahmebereich mit Objekten ausgestattet, die eine ausgeprägte und unverwechselbare Texturierung aufweisen. Im realen Anwendungsfall könnte das zu scannende Objekt diese Anforderung erfüllen, sofern es über ausreichend strukturelle Details verfügt. Für die Versuche wurde der Aufnahmebereich mit Zeitschriften ausgelegt, um eine vielfältige Textur zu gewährleisten.
In jedem Bildpaar werden Schlüsselpunkte erkannt und deren Merkmalsvektoren erstellt. Anschließend werden die Vektoren beider Bilder verglichen und die am besten übereinstimmenden Punkte ermittelt. In diesem Versuch wurden die 32 besten Übereinstimmungen zur Weiterverarbeitung ausgewählt, was der Anzahl der Punkte entspricht, die in den anderen Methoden verwendet werden.

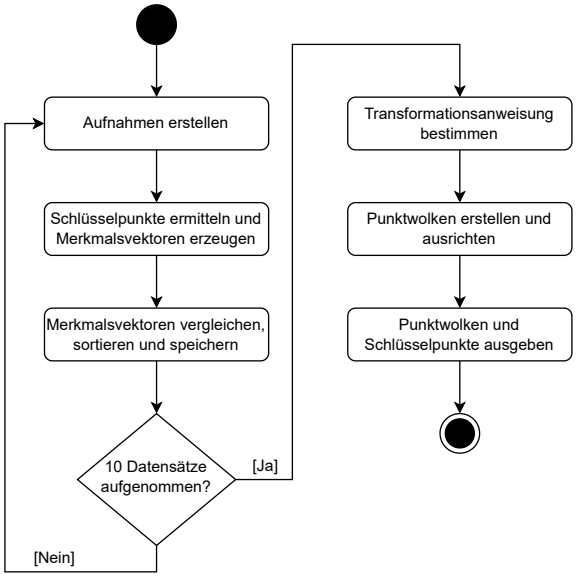


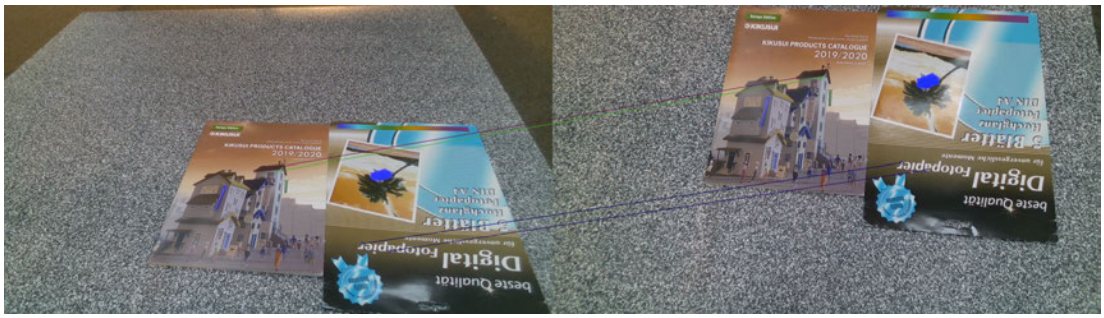
Abbildung 6.12: Ablaufdiagramm der Transformationsbestimmung mit ORB

Für die erkannten Übereinstimmungen werden die X-, Y- und Z-Koordinaten aus den Tiefenbildern extrahiert. Mit den gewonnenen Daten aus den zehn Bildpaaren werden die optimalen Transformationsanweisungen berechnet. In Abbildung 6.12 ist der Ablauf des Prozesses dargestellt.

Ergebnis

Bei den Versuchen mit dem zuvor beschriebenen Algorithmus führten die Ergebnisse nicht zu den erwarteten Erfolgen. In Abbildung 6.13a sind alle im Bild gefundenen Schlüsselpunkte als farbige Kreise dargestellt. Für diesen Versuch wurden pro Bild etwa 10.000 Schlüsselpunkte identifiziert. Beim Zuordnen dieser Punkte wurden etwa 4.000 vermeintlich übereinstimmende Punkte gefunden. In Abbildung 6.13b sind die ersten vier übereinstimmenden Schlüsselpunkte mit Linien verbunden dargestellt. Diese scheinen auf den ersten Blick korrekt zugeordnet zu sein. Eine genauere Überprüfung der vermeintlichen Übereinstimmungen zeigt jedoch, dass ein Großteil der Übereinstimmungen fehlerhaft oder ungenau ist. Selbst bei gut und unterschiedlich texturierten Bildern erweist sich diese Methode als unzureichend, um präzise Ergebnisse zu liefern.

(a) Extrahierte Keypoints: Jede Markierung entspricht einem gefundenen Keypoint



(b) Vermutlich Übereinstimmende Keypoints: Die einander zugeordneten Keypoints sind mit einer Linie verbunden.

Abbildung 6.13: Feature Matching mit ORB

In Abbildung 6.14 sind die vereinten Punktwolken abgebildet. Da die ermittelten Schlüsselpunkte sich in der besten Annäherung auf einer Ebene befinden, ist zu erwarten, dass die berechneten Transformationsanweisungen die Punktwolken auf eine XY-Ebene projizieren. Dieses Verhalten lässt sich in manchen Fällen beobachten, allerdings führen Abweichungen oft zu Verzerrungen, wodurch die Punktwolken nicht einmal auf einer gemeinsamen Ebene liegen.
Die Evaluierungsfunktion zur Überprüfung der Registrierung ergab einen Fitnesswert von etwa 0,17, was einer Überlappung von nur 17 % entspricht. Der RMS-Fehler der überlappenden Punkte ist bei dieser Methode zudem der größte.
Die Verarbeitung eines Datensatzes dauert mit der beschriebenen Konfiguration zwischen 1,5 und 2 Sekunden. Die Berechnung der Transformationsanweisungen für 10 Datensätze (etwa 320 Punkte) erfolgt in wenigen Millisekunden, weswegen diese Dauer vernachlässigbar ist.

Abbildung 6.14: Mit Feature-Matching ausgerichtete Punktwolken. Grüner Pfeil: X-Achse, roter Pfeil: Y-Achse, blauer Pfeil: Z-Achse

6.8.3 Methode 2: detektieren von Markern

Bei dieser Methoden werden Fiducial-Marker zur identifikation von Punkten im Aufnahmebereich verwendet. Fiducial-Marker sind visuelle Referenzmarkierungen, die in der Bildverarbeitung zur Bestimmung der Position und Orientierung von Kameras und Objekten eingesetzt werden. Sie spielen eine zentrale Rolle in Anwendungen wie Augmented Reality, Robotik und automatisierter Inspektion, indem sie als Anhaltspunkte zur präzisen Verfolgung und Lokalisierung dienen [36]. Unter den verschiedenen Fiducial-Marker-Systemen werden ArUco-Marker häufig verwendet, da sie eine einfache Implementierung und robuste Erkennung bieten. Zudem sind sie in Bibliothek OpenCV implementiert.
ArUco-Marker bestehen aus einem binären Code, der in einem quadratischen Raster angeordnet ist. Jeder Marker enthält eine schwarze Umrandung mit einem weißen Innenbereich, in dem sich das spezifische Muster befindet. Dieses Muster ist nicht rotationssymmetrisch, was bedeutet, dass die Marker eine eindeutige Vorderseite und Orientierung besitzen. Dies ermöglicht es nicht nur, ihre Anwesenheit und Position zu erkennen, sondern auch ihre Richtung relativ zur Kamera zu bestimmen. Außerdem lassen sich ArUco-Marker auch aus sehr flachen Winkeln zuverlässig erkennen.

Anwendung

Analog zur vorherigen Methode können bei der Verwendung von ArUco-Markern keine vordefinierten Objektpunkte genutzt werden, da die Marker keine durchgehende Struk-

tur aufweisen. Folglich ist es ohne zusätzlichen Messaufwand nicht möglich, die relative Position der Marker zueinander zu bestimmen.
In jedem der zehn Kalibrierungsbilder werden die Eckpunkte der ArUco-Marker detektiert. Diese Eckpunkte dienen als klare Referenzpunkte, die zuverlässig erkannt werden können. Sobald identische Marker in den Bildern beider Kameras erkannt wurden, werden die entsprechenden Eckpunkte als valide korrespondierende Punkte identifiziert und für die weitere Verarbeitung verwendet. Die zugehörigen X-, Y- und Z-Koordinaten der Marker werden anschließend aus den Tiefenbildern extrahiert. Mithilfe dieser Markerkoordinaten können die optimalen Transformationsparameter berechnet werden, um die Punktwolken der beiden Kameras präzise zueinander auszurichten. Der Ablauf dieser Methode entspricht im Wesentlichen dem der vorherigen. Jedoch werden anstelle von Schlüsselpunkten die Marker genutzt, und zur Erfassung der Kalibrierkoordinaten dienen die Tiefenbilder statt der Farbbilder. Aufgrund des ähnlichen Ablaufs wird an dieser Stelle auf ein weiteres Ablaufdiagramm verzichtet.

Ergebnis

In Abbildung 6.15 sind die ausgerichteten Punktwolken einer Aufnahme dargestellt. Im Gegensatz zur vorherigen Methode sind visuell keine Unstimmigkeiten in der Gesamtpunktwolke erkennbar.
Die Bewertung der Registrierung mithilfe der Open3D-Funktion zeigt, dass etwa 72 % der Punkte überlappen. Zudem weist diese Methode einen geringeren RMS-Fehler auf. Dieser Wert ist kein Garant für eine hohe Präzision der Ausrichtung, wird aber im Hinblick auf die offensichtlich besser ausgerichteten Punktwolken bestätigt.
Die Dauer einer Aufnahme beträgt bei der Verwendung von acht ArUco-Markern etwa 85 Millisekunden. Die Berechnung der Transformationsanweisungen erfolgt, wie auch bei der vorherigen Methoden, in einem vernachlässigbar kurzen Zeitraum, da die gleiche Funktion und eine vergleichbare Anzahl von Punkten verwendet wurde.

6.8.4 Methode 3: Detektieren von Schachbrettmuster

Schachbrettmuster sind ein etabliertes Mittel zur Kalibrierung von Kamerasystemen. Sie bestehen aus abwechselnd schwarzen und weißen quadratischen Feldern, deren Ecken leicht erkannt und lokalisiert werden können, was sie ideal für die Kamerakalibrierung macht. Für eine präzise Kalibrierung muss jedoch das gesamte Muster im Bild erfasst

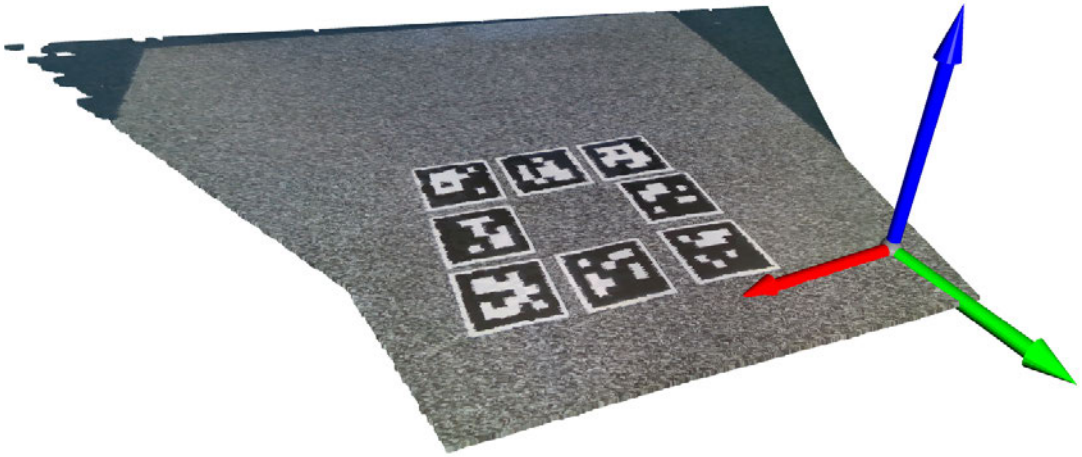


Abbildung 6.15: Mit Hilfe der Detektierung von Markern ausgerichtete Punktwolken. Grüner Pfeil: X-Achse, roter Pfeil: Y-Achse, blauer Pfeil: Z-Achse

sein, damit alle notwendigen Punkte korrekt detektiert werden können.
ArUco-Boards hingegen bestehen aus einzelnen, quadratisch angeordneten ArUco-Markern. Diese Marker bieten den Vorteil einer eindeutigen Zuordnung durch ihr spezifisches Muster, auch wenn Teile des Boards verdeckt sind. Der Nachteil besteht jedoch darin, dass die Eckpositionen der Marker nicht so präzise wie bei einem Schachbrettmuster erkannt werden können.
ChArUco-Boards kombinieren die Vorteile beider Methoden: Die ArUco-Marker dienen als Referenzpunkte, während die Ecken des Schachbrettmusters für die eigentliche Kalibrierung genutzt werden. Dies ermöglicht eine präzisere Bestimmung der Schlüsselpunkte, ohne dass das gesamte Board sichtbar sein muss. [69]. In Abbildung 6.16 ist der Aufbau dieser verschiedenen Boards dargestellt.

Anwendung

Da das ChArUco-Board als ein zusammenhängendes Objekt betrachtet wird, lassen sich die Abstände zwischen den einzelnen Schachbrettpunkten präzise festlegen. Aus diesen definierten Abständen können die entsprechenden 3D-Objektpunkte abgeleitet werden, die die relative Position der Schachbrettecken zueinander beschreiben. Dadurch ist es möglich, die Transformationsanweisungen allein aus den 2D-Bildpunkten zu bestimmen. Für jedes Bildpaar werden dabei die identischen Objektpunkte sowie die entsprechenden 2D-Bildpunkte aus beiden Kameraaufnahmen extrahiert. Auf Grundlage dieser Daten

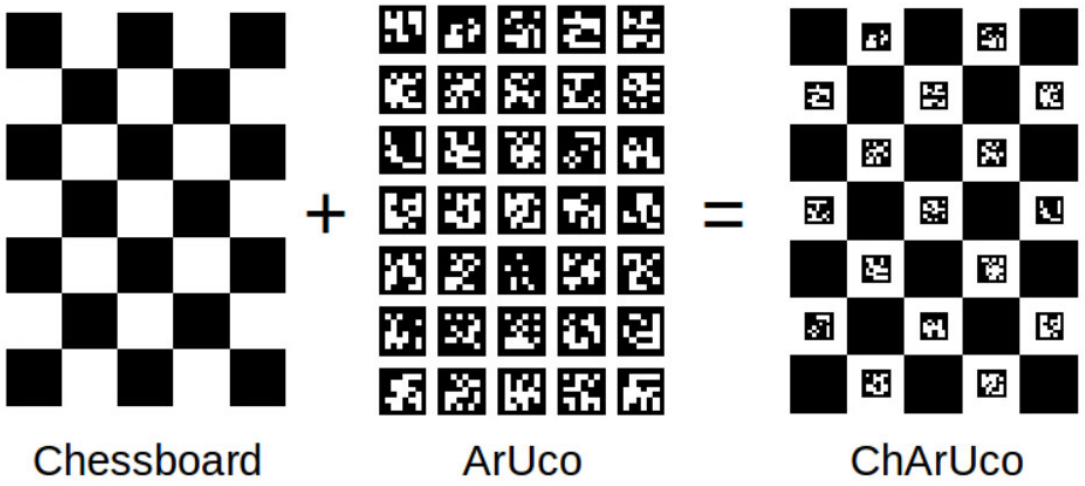


Abbildung 6.16: ChArUco-Boards verbinden die Vorteile von Schachbrettmustern und ArUco-Boards [69]

berechnet die OpenCV-Funktion *stereoCalibrate* anschließend die optimalen Transformationsanweisungen.

Ergebnis

In Abbildung 6.17 sind die zueinander ausgerichteten Punktwolken einer Aufnahme dargestellt. Ähnlich wie bei der Kalibrierung mit den ArUco-Markern zeigen die vereinten Punktwolken keine sichtbaren Unstimmigkeiten.
Die Auswertung mithilfe der Open3D-Funktion bestätigt, dass etwa 73 % der Punkte überlappen. Der RMS-Fehler, der die Genauigkeit der Punktwolkenregistrierung beschreibt, liegt auf einem ähnlichen Niveau wie bei der Kalibrierung mit ArUco-Markern. Die Aufnahmezeit mit einem ChArUco-Board, das 35 Schachbrettfelder enthält und damit 24 Referenzpunkte pro Aufnahme bietet, beträgt etwa 90 Millisekunden. Die Berechnung der Transformationsanweisungen nimmt rund 15 Millisekunden in Anspruch.

6.8.5 Entscheidung

Die Kalibrierung mittels Feature Matching mit dem ORB-Algorithmus wird nicht weiter verfolgt, da die erzielten Ergebnisse im Vergleich zu den beiden anderen Methoden deutlich unterlegen waren. Eine signifikante Verbesserung durch Optimierung erscheint

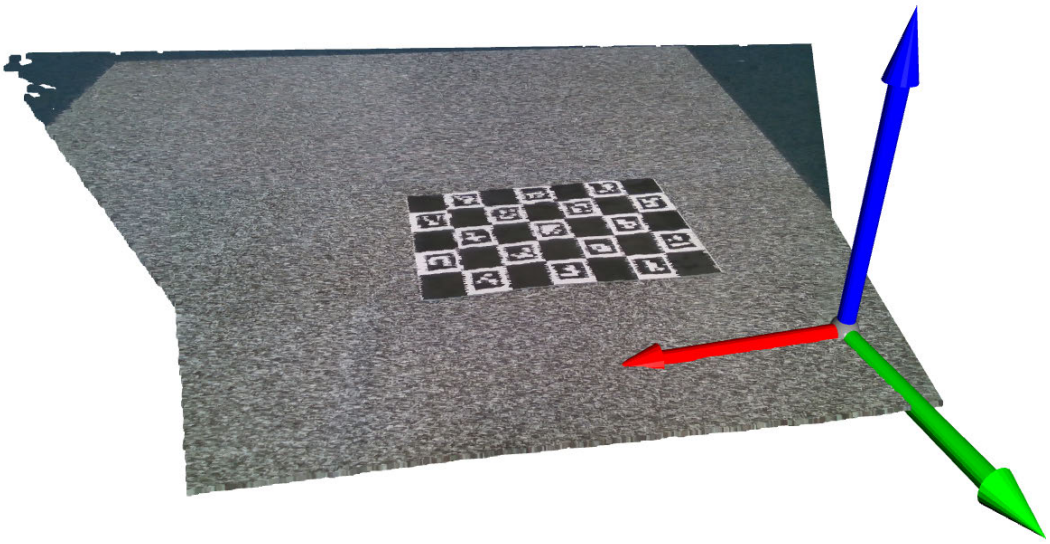


Abbildung 6.17: Ausgerichtete Punktwolken unter der Verwendung von ChArUco-Board

ausgeschlossen.
Für die Kalibrierung der Sensoreinheit vor der Datenaufnahme ist die Effizienz der Berechnung und damit die Ausführungszeit nicht das wichtigste Kriterium. Die erzielte Genauigkeit der Transformationsanweisungen steht im Vordergrund. In diesem Punkt lieferten die Methoden zwei und drei nahezu identische Resultate.
Da beide Methoden hinsichtlich der Kalibrierungsgenauigkeit gleichermaßen geeignet sind, wird die Entscheidung auf Basis der Benutzerfreundlichkeit und Ergonomie getroffen. Die Kalibrierung mit dem ChArUco-Board bietet hierbei den Vorteil, dass bei einer Änderung der Position des Kalibriermusters nur ein einziges Objekt bewegt werden muss, im Gegensatz zu acht separaten ArUco-Markern. Darüber hinaus können die Kalibrieraufnahmen aus einer geringeren Distanz erfolgen, da die Gefahr ungültiger Tiefenwerte nicht gegeben ist. Dies ist besonders bei Szenarien von Vorteil, in denen das Tiefenrauschen, dass sich negativ auf die Genauigkeit der Tiefenmessungen auswirkt, verstärkt auftritt. Unter diesen Gesichtspunkten wird die Kalibrierungsmethode mit dem ChArUco-Board bevorzugt. Als Kalibrierobjekt kann eine Hartschaumplatte mit einem aufgedruckten ChArUco-Muster genutzt werden, die an der Hochschule bereits vorhanden und daher mit keinen Kosten verbunden ist.
Auch bei der Vor- oder globalen Registrierung ist die Genauigkeit von größter Bedeutung. Da die Vorregistrierung während der Datenaufnahme erfolgt, ist in diesem Fall

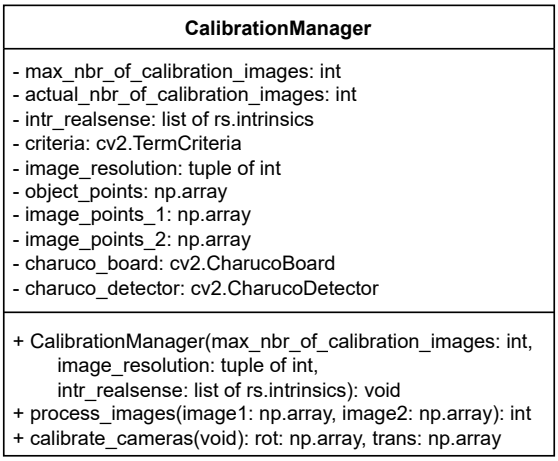


Abbildung 6.18: Klassendiagramm der Klasse *CalibrationManager*

die Ausführungszeit ebenfalls ein entscheidender Faktor. Die Versuche zeigten, dass die Methoden eins und zwei im schlechtesten Fall etwa 105 Millisekunden für die Ausführung benötigen. Dies ermöglicht die Verarbeitung von bis zu neun Datensätzen pro Sekunde. Da bei der Durchführung der Aufnahmen nur alle paar Sekunden ein Bild aufgenommen wird, nachdem der Benutzer eine neue Position eingenommen hat, ist die Verarbeitungsrate ausreichend.
Für die Vorregistrierung wird die Verwendung von separaten ArUco-Markern bevorzugt. Diese haben den Vorteil, dass sie flexibel im Aufnahmebereich platziert werden können, was eine gezielte Begrenzung des Bereichs ermöglicht, in dem sich das zu beobachtende Objekt befindet. Auf diese Weise kann das Objekt präzise erfasst und von der Umgebung getrennt werden.

6.9 Kalibrierung (CalibrationManager)

In der Klasse *CalibrationManager* wird die zuvor beschriebene Methodik implementiert. Konkret werden zwei Methoden implementiert. Mit der einen werden Daten aufgenommen und mit der anderen wird die Transformationsanweisung bestimmt. In Abbildung 6.18 ist das entsprechende Klassendiagramm dargestellt. Im Folgenden werden die Methoden und anschließend die Implementierung ins UI beschrieben. Abschließend werden wieder die Ergebnisse dieses Verarbeitungsschrittes dargestellt.

6.9.1 Konstruktor

Bei der Instanzierung der Klasse werden die Anzahl der gewünschten Kalibrierungsaufnahmen, die gewählte Auflösung der Aufnahmen sowie die intrinsischen Parameter der Kameras übergeben. Daneben werden Attribute initialisiert, die die Handhabung und Verwaltung des ChArUco-Boards ermöglichen. Diese Attribute ermöglichen die Erkennung der Marker und die Zuordnung der Schachbrettpunkte, um die Kalibrieraufnahmen korrekt zu verarbeiten.

6.9.2 Datenaufnahme

Mit der Methoden *process_ images* werden die einzelnen Kalibrieraufnahmen verarbeitet. Übergeben werden dazu die beiden aufgenommenen Farbbilder. In diesen werden die ChArUco-Ecken gesucht. Wenn in beiden Bildern die gleichen Ecken gefunden wurden, werden diese, wie auch die korrespondierenden Objektpunkte, gespeichert. Von der Funktion werden Folgende Werte zurückgegeben:

-1: Weist auf eine ungültige Aufnahme hin, wenn in beiden Bildern verschiedene Marker gefunden wurden.

0: Bestätigt die Aufnahme

1: Wenn die gewünschte Anzahl an Aufnahmen durchgeführt wurde

6.9.3 Transformationsparameter ermitteln

Beim Aufruf der Methode *calibrate_ cameras* wird die Transformationsanweisung berechnet. Zunächst werden die Kameramatrizen basierend auf den übergebenen intrinsischen Parametern erstellt. Diese Kameramatrizen, zusammen mit den erfassten Bildpunkten und den korrespondierenden Objektpunkten, werden dann an die OpenCV-Funktion *stereoCalibrate* übergeben. Die Funktion berechnet die optimale Rotationsmatrix und den Translationsvektor. Diese beiden Ergebnisse werden schließlich zurückgegeben.

6.9.4 Änderungen im UI

Wenn der Benutzer eine Kalibrierung wünscht und die Anzahl der erforderlichen Kalibrieraufnahmen festgelegt wurde, wird der Kalibriervorgang durch die Betätigung der Schaltfläche *Prozess starten* initiiert. In diesem Moment erscheinen zwei neue UI-Elemente: ein Textfeld, das die Anzahl der bereits aufgenommenen Kalibrierbilder anzeigt, sowie eine Schaltfläche, mit der eine weitere Kalibrieraufnahme gemacht werden kann. Gleichzeitig wird durch einen separaten Thread das Livebild der beiden Kameras angezeigt, sodass der Benutzer die Szene verfolgen kann.
Betätigt der Benutzer die Schaltfläche, wird der Thread, der das Livebild aktualisiert, kurzzeitig gestoppt, um die aktuellen Bilder zu erfassen und zu speichern. Nach der erfolgreichen Aufnahme wird der Livebild-Stream wieder gestartet, und der Prozess wartet auf die nächste Aufnahme.
Sobald die erforderliche Anzahl an Aufnahmen erreicht ist, werden die Transformationsanweisungen berechnet und in einer Datei gespeichert, um sie für zukünftige Aufnahmen verwenden zu können. Abschließend wird der Kalibrierprozess beendet, indem die hinzugefügten UI-Elemente entfernt werden. Anschließend kann der eigentliche Prozess der Datenaufnahme beginnen.

6.9.5 Ergebnis

In Abbildung 6.19 sind die beiden zueinander ausgerichteten Punktwolken einer Aufnahme dargestellt. Aus dieser Perspektive ist kein Übergang zwischen den beiden Teilpunktwolken erkennbar. Erst bei sehr genauer Betrachtung, bei der die einzelnen Punkte sichtbar werden, lässt sich ein leichter Versatz feststellen. Dieser Versatz wird bei der finalen Verarbeitung durch einen zusätzlichen Durchgang der lokalen Registrierung korrigiert werden müssen, um ein optimales Ergebnis zu erzielen.

6.10 Datenaufnahme (DatasetManager)

Die Datenaufnahme wird in der Klasse *DatasetManager* organisiert. Diese Klasse enthält alle notwendigen Methoden, um die Datensätze sequenziell aufzunehmen und zu verarbeiten. Dabei wird jeder Datensatz nacheinander aufgenommen, global vorregistriert und anschließend gespeichert. Zur globalen Registrierung wird, wie bereits im

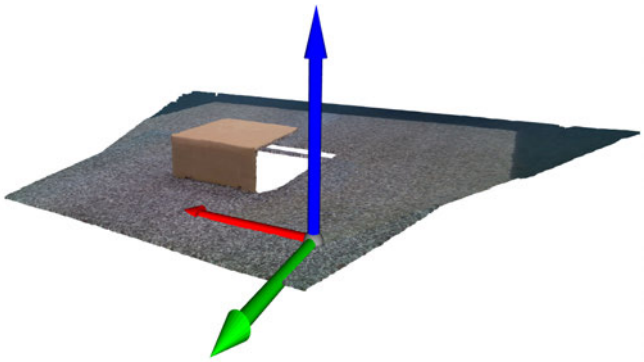


Abbildung 6.19: Zueinander ausgerichtete Punktwolken beider Kameras

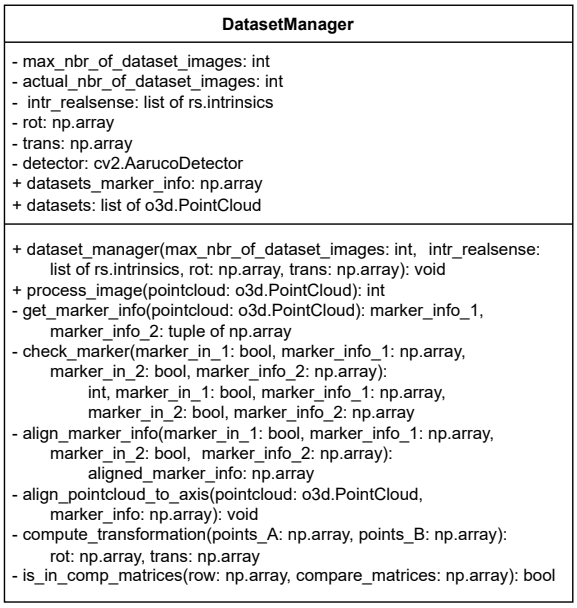


Abbildung 6.20: Klassendiagramm der Klasse *DatasetManager*

vorherigen Kapitel beschrieben, eine auf ArUco-Markern basierende Methode verwendet. Diese Ausrichtung erfolgt zur Laufzeit der Datenaufnahme, sodass die Daten anschließend direkt weiterverarbeitet werden können. Um einen unproblematischen Prozessablauf zu unterstützen, wurden acht Hartschaumplatten bedruckt mit jeweils einem ArUco-Marker bestellt. Diese haben das Budget mit 17,84 € belastet. In Abbildung 6.20 ist das Klassendiagramm dargestellt.

6.10.1 Konstruktor

Ein Objekt der Klasse *DatasetManager* wird instanziiert, indem es die Anzahl der gewünschten Aufnahmen, die intrinsischen Parameter der Kameras sowie die Transformationsanweisungen, die aus einer vorherigen Kalibrierung resultiert sind, übergeben bekommt. Beim Aufruf des Konstruktors werden diese Informationen als Attribute des Objekts gespeichert. Zusätzlich wird der Detektor zur Identifikation der ArUco-Marker initialisiert, mit dem die Erkennung der Marker in den aufgenommenen Bildern ermöglicht wird. Des Weiteren wird ein Array angelegt, in dem die Koordinaten der identifizierten ArUco-Marker sowie die zugehörigen IDs für alle Aufnahmen gespeichert werden, um eine Ausrichtung der aufgenommenen Datensätze zu gewährleisten.

6.10.2 Datenaufnahme

Mit der Methode *process_image* werden einzelne Datensätze aufgenommen. Der Ablauf dieser Methode ist in Abbildung 6.21 visualisiert. Es wird ein Objekt der Klasse *PointCloud* übergeben. Zunächst werden mithilfe der Funktion *get_marker_info* die ArUco-Marker in den Farbbildern identifiziert und deren Informationen extrahiert. Diese Funktion liefert für jedes Farbbild ein Array zurück, das die Koordinaten der erkannten Markereckpunkte enthält. Zusätzlich werden die zugehörigen Marker-IDs und die Eckpunktnummern (von eins bis vier) für jede Ecke bereitgestellt.
Anschließend werden die detektierten Marker mit der Funktion *check_marker* überprüft. Dabei werden zunächst alle Einträge entfernt, deren Koordinaten ungültig Werte enthalten (X=Y=Z=0). Danach wird geprüft, ob in den Bildern beider Kameras zusammen mindestens drei valide Markerecken vorhanden sind. Dies ist erforderlich, um eine gültige Transformationsmatrix zu berechnen, die den aktuellen Datensatz an die anderen Datensätze ausrichtet.
Ist diese Prüfung erfolgreich und das Bild somit gültig, wird ermittelt, ob es sich um den ersten Datensatz handelt. Falls ja, werden die Punktwolken des aktuellen Datensatzes erstellt und ausgerichtet. Anschließend werden sowohl die Punktwolken als auch die Markerkoordinaten mithilfe der Funktion *align_pointcloud_to_axis* so transformiert, dass sie auf der XY-Ebene bei Z = 0 liegen. Die Markerkoordinaten werden in einem Array gespeichert, das die Markerinformationen aller Datensätze vereint. Der verarbeitete Datensatz wird ebenfalls gespeichert.

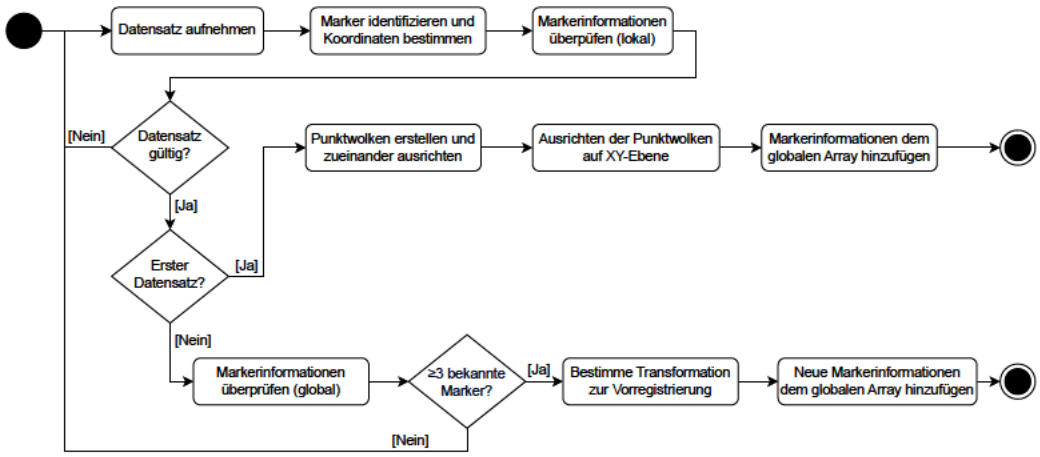


Abbildung 6.21: Ablaufdiagramm der Methode *process_image*

Handelt es sich nicht um den ersten Datensatz, wird geprüft, ob das Array mit den gesammelten Markerinformationen mindestens drei Markerecken enthält, die im aktuellen Datensatz erkannt wurden. Ist dies der Fall, wird die Funktion *compute_transformation* aufgerufen, um die Transformationsparameter zu berechnen und die aktuelle Aufnahme an die vorherigen Datensätze anzupassen. Neu erkannte Markerkoordinaten aus dieser Aufnahme werden dem bestehenden Array hinzugefügt.

6.10.3 Änderungen im UI

Für diesen Prozess werden ähnliche Änderungen wie bei der Kalibrierung vorgenommen. Wenn das Live-Bild noch nicht aktiviert ist, wird der entsprechende Thread gestartet, um die Live-Bilder der Kameras anzuzeigen. Zur Steuerung der Datenaufnahme wird eine Schaltfläche hinzugefügt, mit der der Benutzer die Aufnahme eines Datensatzes manuell starten kann, sowie ein Textfeld, das die Anzahl der bereits aufgenommenen Datensätze anzeigt.
Sobald der Benutzer die Schaltfläche betätigt, wird ein neuer Datensatz aufgenommen, der automatisch entweder ausgerichtet oder, falls es sich nicht um den ersten Datensatz handelt, zu den vorherigen Datensätzen ausgerichtet wird. Nachdem die vom Benutzer festgelegte Anzahl an Datensätzen erfolgreich aufgenommen wurde, werden die hinzugefügten UI-Elemente wieder entfernt und der Thread, der das Live-Bild anzeigt, wird gestoppt.

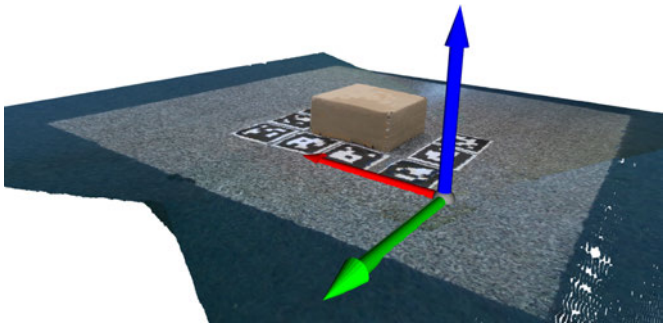


Abbildung 6.22: Zueinander ausgerichtete Punktwolken beider Kameras

6.10.4 Ergebnis

In Abbildung 6.22 sind die zueinander ausgerichteten Punktwolken von vier Datensätzen dargestellt. Der erste Datensatz wurde ohne Objekt aufgenommen, um alle verfügbaren Marker zu erfassen. Im zweiten Datensatz wurde das Objekt zwischen den Markern platziert. Bei den weiteren Datensätzen wurde die Kameraposition verändert, um Aufnahmen aus unterschiedlichen Perspektiven zu erhalten.
Die Aufnahmen wurden erfolgreich zu einer Gesamtpunktwolke zusammengefügt, die anschließend mit einer lokalen Registrierung hätte optimiert werden können. Allerdings führte die große Menge an überwiegend unnötigen Punkten zu einer merklich längeren Verarbeitungszeit.

6.11 Vorverarbeitung (Preprocessor)

Nachdem die Daten aufgenommen wurden, müssen diese vorverarbeitet werden, damit im Nachhinein die lokale Registrierung durchgeführt werden kann. Das Ziel dabei ist es, die Daten so vorzuverarbeiten, dass die Registrierung effizient, also in einem angemessenen Zeitraum durchgeführt werden kann und zu einem möglichst genauen Ergebnis führt. Nach Anforderung **SW-GE 5** soll die Verarbeitung für eine Perspektive bei maximal einer Minute liegen um eine Punktwolke mit einer Abweichung von ±0,2 mm zu erzeugen (**SW-GE 2**). Um das zu erreichen werden im Folgenden zunächst die Punktdaten extrahiert, die dem beobachteten Objekt zugeordnet werden können. Anschließend werden diese Punkte einer Filterung unterzogen, die außenliegende Punkte und Bereiche mit einer zu geringen Punktdichte entfernt.

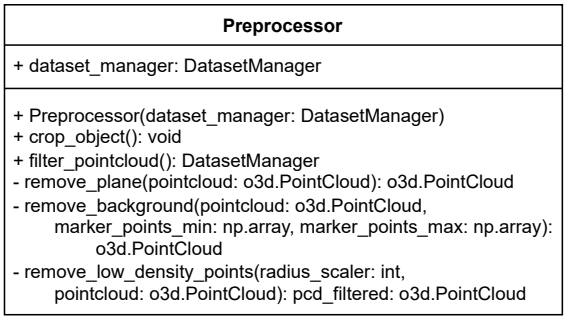


Abbildung 6.23: Klassendiagramm der Klasse *Preprocessor*

Die Methoden werden in der Klasse *Preprocessor* implementiert. Das Klassendiagramm dieser Klasse ist in Abbildung 6.23 dargestellt.

6.11.1 Konstruktor

Bei der Instanziierung wird ein Objekt des Typs *DatasetManager* übergeben, welches im Konstruktor als Attribut gespeichert wird.

6.11.2 Objekt ausschneiden

Die folgenden Schritte zur Isolierung des Objekts werden in der Methode *crop_capture_area_and_background* durchgeführt.
Im ersten Schritt der Vorverarbeitung werden die Punkte entfernt, die der Grundfläche des Aufnahmebereichs zugeordnet werden können, also die Punkte, die auf dem ausgelegten Rauschmuster liegen. Um diese Punkte zu identifizieren, wird aus den Punktwolken die dominierende Ebene bestimmt, die der Fläche des Aufnahmebereichs entspricht. Dies setzt voraus, dass das Objekt selbst keine größeren Flächen aufweist und der Hintergrund im Verhältnis zum Aufnahmebereich keine überproportional großen Flächen enthält.
Die Identifikation der dominierenden Ebene erfolgt mithilfe der Open3D-Funktion *segment_plane*, deren detaillierte Beschreibung sich in Abschnitt 6.2.1 befindet. Die Funktion liefert die Indizes der Punkte zurück, die der gefundenen Ebene zugeordnet werden können, welche anschließend aus den Punktwolken entfernt werden.
In Abbildung 6.24 ist das Ergebnis am Beispiel einer einzelnen Punktwolke dargestellt.

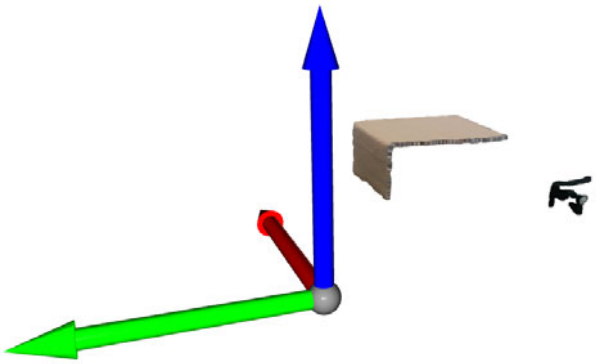


Abbildung 6.24: Punktwolke ohne Aufnahmebereich

In diesem Fall wurde das Objekt auf ein kleines Podest gestellt, um sicherzustellen, dass der untere Bereich des Objekts während des Ausschneideprozesses nicht abgeschnitten wird. Zusätzlich wurde im Hintergrund ein Objekt platziert, das nicht Teil der Aufnahme sein soll und stellvertretend für mögliche unerwünschte Objekte im Hintergrund oder unterhalb des Aufnahmebereichs steht. Diese unerwünschten Elemente werden im nächsten Schritt entfernt.

Nachdem mit der zuvor beschriebenen Methode bereits ein Großteil der unerwünschten Punkte entfernt wurde, erfolgt im nächsten Schritt die Beseitigung der verbleibenden Störpunkte, die im ersten Durchgang nicht erfasst wurden. Dazu zählen beispielsweise Punkte von Objekten im Hintergrund, die sich oberhalb oder unterhalb der dominierenden Ebene des Aufnahmebereichs befinden.
Für diesen Prozess werden zunächst die minimalen und maximalen X- und Y-Koordinaten aus dem Array ermittelt, das alle identifizierten ArUco-Markerecken enthält. Anschließend werden die Punktkoordinaten sowie die zugehörigen Farbwerte extrahiert und in separaten Arrays gespeichert. Mithilfe dieser Koordinaten wird eine Maske erstellt, die alle Punkte umfasst, die innerhalb der ermittelten minimalen und maximalen Markerkoordinaten liegen. Auf dieser Basis wird eine neue Punktwolke generiert, die nur noch die relevanten Punkte und die korrespondierenden Farbwerte enthält. Es wird dabei angenommen, dass sich innerhalb der platzierten Marker ausschließlich das zu scannende Objekt befindet.
In Abbildung 6.25 ist das Ergebnis dieses Schritts dargestellt. Alle Punkte, die zu Objekten im Hintergrund gehörten, wurden erfolgreich entfernt. Die Reduktion der Punktanzahl verdeutlicht die Effizienz dieser Vorgehensweise. Eine ursprüngliche Punktwolke

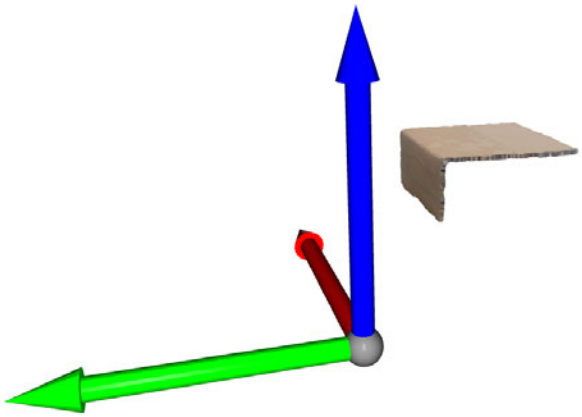


Abbildung 6.25: Punktwolke ohne Hintergrund

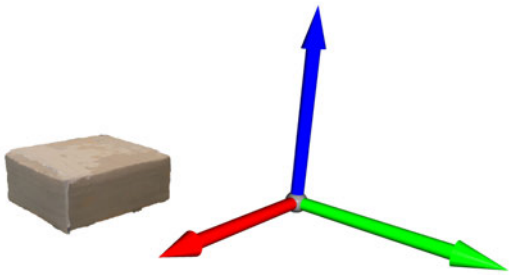
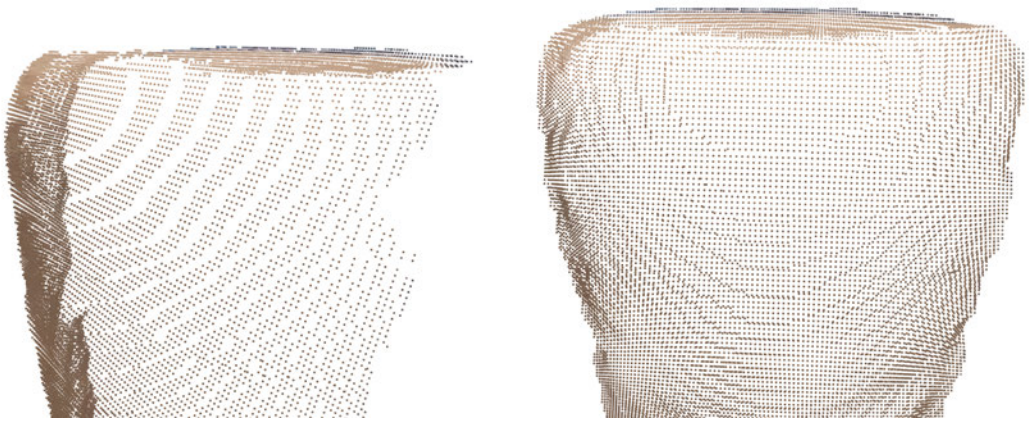


Abbildung 6.26: Ausgeschnittene Punktwolken der zusammengefügten Datensätze

umfasste etwa 800.000 Punkte. Nach dem Entfernen der Punkte, die der Grundfläche zugeordnet waren, verbleiben rund 80.000 Punkte, die dem zu erfassenden Objekt sowie dem Hintergrundobjekt zugeordnet werden können. Durch das Ausschneiden des Hintergrundobjekts wird die Punktwolke auf etwa 77.000 Punkte reduziert. Dies entspricht einer Verringerung der Punktanzahl um den Faktor 10, wodurch die nachfolgenden Verarbeitungsschritte erheblich effizienter durchgeführt werden können.

Ergebnis: Isolierung des Objekts

In Abbildung 6.26 sind die zusammengefügten Punktwolken aus vier Datensätzen dargestellt, die schon in den vorherigen Kapiteln gezeigt wurden. Der Aufnahmebereich und der Hintergrund wurde für jede Teilpunktwolke erfolgreich ausgeschnitten.



(a) Geringe Punktdichte: Oberfläche in spitzem Winkel aufgenommen

(b) Hohe Punktdichte: Objekt aus 90°-Winkel aufgenommen

Abbildung 6.27: Punktdichten abhängig vom Winkel der Oberfläche zur Sichtachse

6.11.3 Filterung

Oberflächen, die aus einem spitzen Winkel aufgenommen werden, können aufgrund der Perspektive eine verringerte Punktdichte aufweisen. Je spitzer der Winkel, desto stärker macht sich die begrenzte Tiefenauflösung bemerkbar. Wenn eine Fläche nahezu parallel zur Sichtachse der Kamera liegt, wird dieser Effekt besonders deutlich. In solchen Fällen zeigen die erfassten Punkte aus Sicht der Kamera nur geringe Veränderungen in der X- und Y-Richtung, während sie in der Tiefe (Z-Richtung) relativ stark variieren. Dies führt zu einem feinen, regelmäßigen Treppenmuster in der Punktwolke. Um diese Verzerrungen zu minimieren, sollten solche Bereiche aus einem anderen Blickwinkel aufgenommen werden, bei dem die Sichtachse der Kamera in einem stumpferen Winkel auf die Oberfläche trifft. Dieser Effekt ist besonders bei runden Objekten ausgeprägt, da sich der Aufnahmewinkel entlang der gekrümmten Oberfläche kontinuierlich ändert, bis die Kamera die Oberfläche aufgrund von Abschattung nicht mehr erfassen kann. Abbildung 6.27 veranschaulicht diesen Effekt anhand der Aufnahme eines runden Objekts. Um die Genauigkeit der Punktwolke zu erhöhen, sollten Punkte, die in solchen problematischen Bereichen liegen, aus der aktuellen Perspektive entfernt werden.

Ein anderes Problem tritt bei Objektkanten auf, die sich am von der Kamera abgewandten Rand des Objekts befinden. Hier kann es zu Unregelmäßigkeiten kommen, da der Übergang von der hinteren Objektkante zum Hintergrund im zweidimensionalen Bild nur wenige Pixel umfasst. Dies führt zu einer fehlerhaften Erkennung der Kanten und

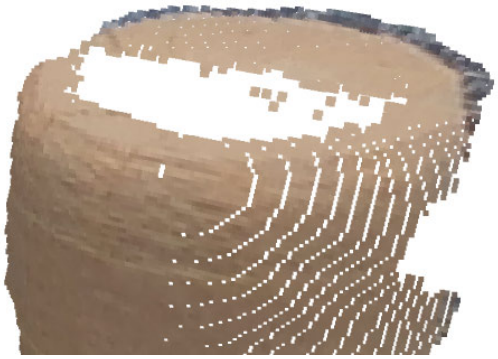


Abbildung 6.28: Hintere Kante erhält Textur des Hintergrunds

kann dazu führen, dass die Hintergrundtextur irrtümlich den Punkten an der hinteren Kante des Objekts zugeordnet wird. Der zweite beschriebene Effekt lässt sich besonders gut bei stumpfen Kanten an der Oberseite eines Objekts beobachten. In Abbildung 6.28 ist dasselbe runde Objekt aus einer Perspektive zu sehen, die die Oberseite zeigt. An der von der Kamera abgewandten Seite nehmen die Objektpunkte die Textur des Hintergrunds an. In diesem Fall die Textur des ausgelegten Rauschmusters. Diese Punkte sollten ebenfalls entfernt werden, da sie die Genauigkeit der späteren lokalen Registrierung beeinträchtigen und zu einer falschen Wiedergabe der Objekttextur führen.

Filtermethodik

Die zuvor beschriebenen Punkte beider Gruppen weisen gemeinsam auf globaler Ebene eine geringere Punktdichte auf als die regulären, qualitativ hochwertigen Bereiche. Dies betrifft sowohl Punkte in Bereichen mit ohnehin niedriger Punktdichte als auch Punkte, denen fälschlicherweise eine falsche Textur zugewiesen wurde, weil sie sich am Rand der Punktwolke befinden. Solche Bereiche können durch eine Analyse der lokalen Punktdichte identifiziert werden. Da es sich um eine unstrukturierte Punktwolke handelt, ist dieser Prozess jedoch rechenintensiver als bei strukturierten Daten wie zweidimensionalen Bildern.
Um die Dichte der Punktwolke zu ermitteln und unerwünschte Punkte zu entfernen, müssen benachbarte Punkte identifiziert und die Abstände zwischen den nächstgelegenen Punkten bestimmt werden. Die Grundlage für diesen Prozess bildet ein K-d-Baum, der es ermöglicht, benachbarte Punkte effizient zu bestimmen. Wie im Grundlagenka-

pitel beschrieben, organisiert der K-d-Baum die Daten so, dass Suchanfragen, beispielsweise nach den nächstgelegenen Punkten, mit geringem Rechenaufwand durchgeführt werden können. Der Filterungsprozess besteht aus folgenden Schritten:

1. **Bestimmung der nächstgelegenen Nachbarn:** Mithilfe des K-d-Baum wird für jeden Punkt der Abstand zu seinem nächstgelegenen Nachbarn ermittelt.

2. **Festlegung des Suchradius:** Der maximal ermittelte Abstand wird skaliert und als Radius r für den nächsten Schritt verwendet.

3. **Berechnung der Nachbaranzahl:** Für jeden Punkt wird die Anzahl der Nachbarpunkte innerhalb des Radius r bestimmt.

4. **Statistische Analyse:** Der Mittelwert μ und die Standardabweichung σ der Anzahl der Nachbarpunkte werden berechnet.

5. **Identifikation und Entfernung von Ausreißern:** Punkte, deren Anzahl an Nachbarn mehr als eine Standardabweichung unter dem Mittelwert liegt (also weniger als $\mu - \sigma$ Nachbarn haben), werden als Ausreißer klassifiziert und entfernt. In einer normalverteilten Punktwolke entspricht dies etwa den 16 % der Punkte mit der geringsten Nachbaranzahl.

Dieser Filtervorgang wird in zwei Durchläufen durchgeführt: Zunächst mit einem kleinen Radius, um Bereiche mit geringer Punktdichte zu entfernen, und anschließend mit einem größeren Radius, um weit entfernte Ausreißer oder unregelmäßige Punktansammlungen zu eliminieren. Für die Umsetzung dieses Prozesses werden zwei Ansätze betrachtet, die sich in ihrer Implementierung geringfügig unterscheiden:

- **Ansatz mit Open3D:** Die erste Methode verwendet die Open3D-Bibliothek zur Erstellung des K-d-Baum. Der Vorteil liegt in der einfachen Integration, da Open3D bereits an vielen anderen Stellen des Projekts verwendet wird. Der Nachteil besteht darin, dass Suchvorgänge wie die Ermittlung der nächstgelegenen Nachbarn nur für einzelne Punkte durchgeführt werden können, was eine Schleifenimplementierung erfordert und die Performance beeinträchtigen kann.

- **Ansatz mit SciPy:** Die zweite Methode basiert auf der SciPy-Bibliothek, die den Vorteil bietet, K-d-Baum-Suchvorgänge parallel auf mehreren Prozessorkernen durchzuführen. Zudem können SciPy Suchprozesse mit der Gesamtzahl der

Punkte gleichzeitig durchgeführt werden, was zu einer deutlichen Verbesserung der Performance führen kann.

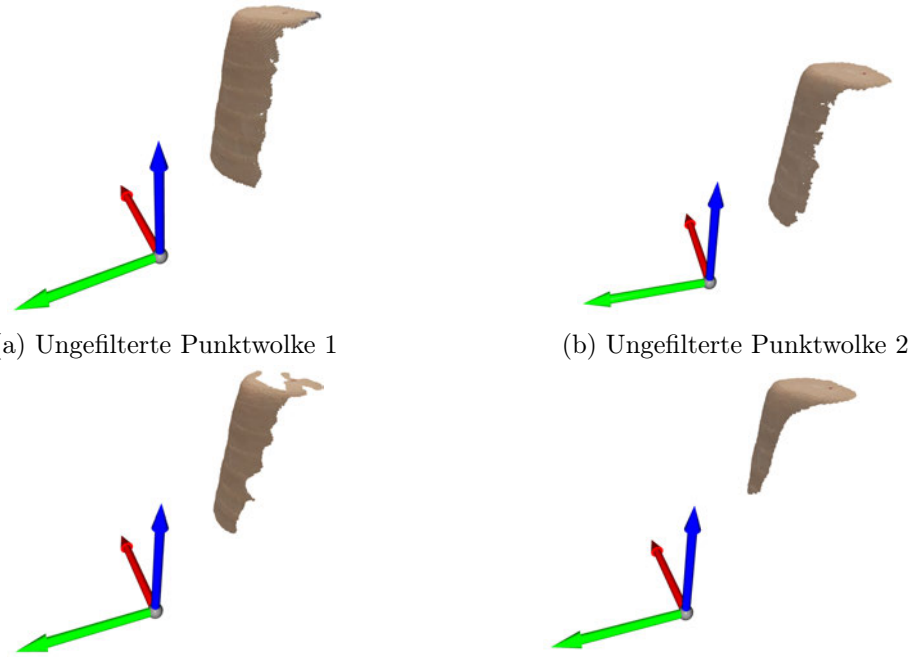
Der Vergleich beider Methoden konzentriert sich hauptsächlich auf die Performance, um die effizienteste Lösung für die Filterung der Punktwolke zu identifizieren.

Ergebnis: Gefilterte Objektpunkte

Die Ergebnisse, die die beiden Algorithmen erzeugen, sind nahezu identisch. Zur Verdeutlichung der Ergebnisse sind diese in Abbildung 6.29 dargestellt. Abbildung 6.29a und 6.29b zeigen die ungefilterten aber ausgeschnittenen Punktwolken. In Abbildung 6.29c und 6.29d sind die beiden Teilpunktwolken nach der Filterung dargestellt. Abschließend ist in Abbildung 6.30 das Endergebnis der Filterung dargestellt, in dem beide gefilterten Punktwolke in einer Punktwolke vereint sind.
Während die Qualität der Filterung bei beiden Methoden gleich ist, unterscheiden diese sich bei der Ausführungszeit erheblich. Die Filterung unter Nutzung der Open3D-Methoden benötigt im Schnitt fünf Sekunden um die zwei Punktwolken zu filtern. Unter Nutzung der SciPy-Methoden ist der Vorgang im Schnitt in etwa 700 Millisekunden abgeschlossen. Damit wird entschieden, dass die Filterung mit den SciPy-Methoden durchgeführt wird.

Umsetzung

Die Punktwolkenfilterung wird in der Methode *filter_pointcloud* durchgeführt. Für jeden Datensatz werden die beiden Teilpunktwolken ausgelesen und mit der Methode *remove_low_density_points* gefiltert. In dieser Methode wird pro Filterdurchgang ein K-d-Baum generiert. Anschließend wird für jeden Punkt der Abstand zu seinem nächsten Nachbarn und von diesen Werten der maximale Wert ermittelt. Dieser Abstand wird anschließend skaliert, wobei der Skalierungsfaktor vom Benutzer individuell auf die jeweilige Szenerie abgestimmt wird. Nach der Skalierung werden die Punkte ermittelt, die eine zu geringe Punktdichte aufweisen, und diese aus der Punktwolke entfernt



(a) Ungefilterte Punktwolke 1 (b) Ungefilterte Punktwolke 2

(c) Gefilterte Punktwolke 1 (d) Gefilterte Punktwolke 2

Abbildung 6.29: Filtervorgang Ergebnis

6.11.4 Änderungen im UI

Nach Abschluss der Datenaufnahme und Beendigung des Livestreams wird automatisch das UI zur Vorverarbeitung angezeigt. Dieses fügt die 3D-Ansicht der ersten aufgenommenen Punktwolke sowie zwei Schieberegler und zwei Schaltflächen hinzu. Die Schieberegler ermöglichen es dem Benutzer, die Skalierungsfaktoren für die Punktwolkenfilterung individuell einzustellen. Mit der Schaltfläche *Punktwolke aktualisieren* wird die angezeigte Punktwolke basierend auf den aktuellen Einstellungen gefiltert und neu dargestellt. Wenn das Filterergebnis den Anforderungen des Benutzers entspricht, kann die Filterung durch Drücken der Schaltfläche *Skalierungswerte übernehmen* mit den aktuellen Werten auf alle Datensätze angewendet werden. Nach Abschluss der Filterung werden die Punktwolkenanzeige sowie die neu hinzugefügten UI-Elemente entfernt. In Abbildung 6.31 ist das UI an diesem Schritt dargestellt.

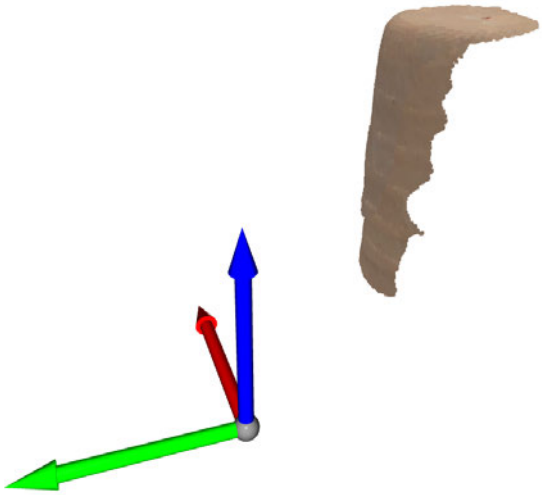


Abbildung 6.30: Zusammengefügte und gefilterte Punktwolken

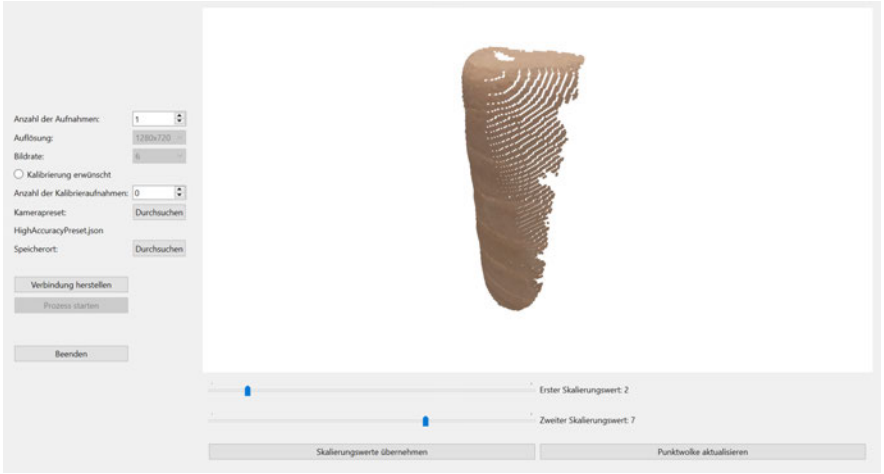


Abbildung 6.31: Benutzerschnittstelle Filterung

6.12 Lokale Registrierung

Im finalen Schritt der Verarbeitungskette erfolgt die lokale Registrierung der Daten. Dieser Abschnitt beschreibt, wie die Punktwolken präzise zueinander ausgerichtet werden. Der Prozess gliedert sich in zwei Stufen: Zunächst werden die durch die Kalibrierung der Sensoreinheit vorregistrierten Punktwolkenpaare final angeglichen und zu einer einzelnen Punktwolke zusammengefasst. Anschließend werden die verschiedenen Punkt-

wolken, die gemeinsam das gesamte Objekt repräsentieren, zueinander ausgerichtet und zu einer Gesamtpunktwolke fusioniert.

6.12.1 Registrierung der Punktwolkenpaare

Die Registrierung der Punktwolkenpaare ist erforderlich, da die Ausrichtung nach der Kalibrierung der Sensoreinheit nicht perfekt ist und zu leichten Abweichungen führt. Für die finale Anpassung wird der ICP-Algorithmus verwendet, wobei ein Ansatz nach [91] zum Einsatz kommt. Dieser folgt einem Grob-zu-Fein-Ansatz, bei dem der Registrierungsalgorithmus iterativ auf verschiedenen Auflösungsebenen angewendet wird.
In jeder Iteration werden die Punktwolken durch Voxelisierung auf unterschiedlichen Auflösungen repräsentiert. Dabei werden Punkte innerhalb eines bestimmten Bereichs, definiert durch die Voxelgröße, zu einem repräsentativen Punkt zusammengefasst, der die lokale Nachbarschaft beschreibt. Zunächst werden große Voxel verwendet, die in jeder Iteration verkleinert werden. Parallel dazu wird die maximale Korrespondenzdistanz reduziert. In den ersten Iterationen mit großen Voxeln wird eine größere Korrespondenzdistanz genutzt, um die größeren Abstände zwischen den Punkten zu berücksichtigen. In späteren Iterationen mit kleineren Voxeln kann die Korrespondenzdistanz verringert werden, da die Punktdichte steigt und die Abstände zwischen den Punkten geringer werden.
Die in jeder Iteration ermittelte Transformationsmatrix wird als Anfangsschätzung für die folgende Iteration verwendet. Dadurch wird die optimale Transformation schrittweise angenähert, was dem grundlegenden Prinzip des ICP-Algorithmus entspricht. Die Abbruchkriterien, die die Verbesserung zwischen den Iterationen beschreiben, werden strenger angesetzt, je höher die Auflösung der Punktwolke ist.
Gemäß [91] führt dieser multiskalare Ansatz zu einem Zeitgewinn gegenüber dem herkömmlichen ICP-Algorithmus. Bei den in dieser Arbeit verwendeten Punktwolken, die aufgrund von Rauschen eine unregelmäßige Objektoberfläche aufweisen, reduziert die Methode zudem das Risiko, in lokalen Minima zu konvergieren, aus denen der Algorithmus nicht mehr herausfindet. Der Grob-zu-Fein-Ansatz kann außerdem den Einfluss einer suboptimalen initialen Ausrichtung abschwächen.
Die Implementierung dieser Registrierung erfolgt mithilfe der Open3D-Methode `multi_scale_icp`. Der Quellcode und die Implementierungsdetails sind in [67] verfügbar. Im Folgenden wird der Ablauf der Methode mit den verwendeten Strategien beschrieben:

- **Vorverarbeitung der Punktwolken:** Beide Punktwolken werden entsprechend den Vorgaben herunterskaliert und gespeichert. Die Voxelgröße wird als Vielfaches des mittleren Abstands aller Punkte zu ihren nächsten Nachbarn definiert. Die Registrierung beginnt mit der geringsten Auflösung.

- **Konstruktion von k-d-Bäumen:** Für beide Punktwolken wird ein K-d-Baum erstellt, um die Nachbarschaftsanalyse zu ermöglichen.

- **Anwendung des ICP-Algorithmus:** Der ICP-Algorithmus wird auf die herunterskalierten Datensätze angewendet, bis eines der Konvergenzkriterien erfüllt ist. Für die erste grobe Iteration werden die Kriterien so gewählt, dass eine hohe relative Änderung erwartet wird und viele Iterationen durchgeführt werden. Die Konvergenzkriterien umfassen:

  - **Relativer Fitnesswert:** Gibt den Grad der Überlappung der Punktwolken im Vergleich zur vorherigen Iteration an.

  - **Relativer RMS-Fehler:** Entspricht der Änderung des durchschnittlichen Abstands korrespondierender Punkte zwischen den Iterationen.

  - **Maximale Anzahl von Iterationen:** Begrenzt die Anzahl der Iterationen des Algorithmus.

- **Korrespondenzsuche:** In jeder Iteration wird für jeden Punkt der Ausgangspunktwolke der nächstgelegene Punkt in der Zielpunktwolke ermittelt, sofern dieser innerhalb der maximalen Korrespondenzdistanz liegt. Es wird die Standardmethode der nächstgelegenen Punkte (*Nearest Neighbor Search*) verwendet (siehe Abschnitt 2.3.2). Eine spezielle Punktauswahl erfolgt nicht. Durch die Voxelisierung können alle Punkte für die Korrespondenzsuche genutzt werden. In späteren Iterationen wird die Korrespondenzdistanz verringert, um die Suche auf relevante Punkte zu beschränken.

- **Transformationsermittlung:** Bei gefundenen Korrespondenzen innerhalb der zulässigen Distanz wird die optimale Transformation basierend auf der vorgegebenen Fehlermetrik berechnet. Korrespondenzen werden nicht weiter bearbeitet; es werden keine Paare verworfen oder unterschiedlich gewichtet.

- **Iterativer Prozess:** Die Schritte werden für jede Auflösungsebene wiederholt, wobei die Punktwolken sukzessive in höherer Auflösung verarbeitet werden. Der

Prozess endet, wenn auf der feinsten Ebene eines der Konvergenzkriterien erfüllt ist.

Die mit dieser Methode erzielten Ergebnisse sind insgesamt zufriedenstellend. Trotz des vorhandenen Rauschens werden die Punktwolken erfolgreich ausgerichtet und konvergieren zu einer Lösung. Die aktuelle Open3D-Implementierung verwendet keine fortgeschrittenen Ansätze, wie sie in Kapitel 2.3.2 beschrieben sind. Durch Anpassungen oder die Implementierung eigener Methoden könnten Genauigkeit und Konvergenzgeschwindigkeit weiter verbessert werden.

Ergebnis

Während der Entwicklung und insbesondere beim Testen traten Einschränkungen auf, die hauptsächlich vom betrachteten Objekt abhängen. Während für die Aufnahme eine matte und helle Oberfläche ideal ist, ergeben sich für die Registrierung zusätzliche Anforderungen.
Abbildung 6.32 zeigt das Registrierungsergebnis eines quaderförmigen Testobjekts. Da die Aufnahmen von einer Seitenfläche aus erfolgten, stellen die beiden Teilbilder lediglich diese Fläche und einen Teil der Deckfläche dar. In dieser Ausrichtung können nur fünf der sechs Freiheitsgrade der Transformation gesperrt werden, was zu einer schlechten Registrierung führt. Geometrisch sind die Punktwolken gut angenähert, doch anhand der Textur wird deutlich, dass sie nicht optimal ausgerichtet sind. Dieses Verhalten resultiert aus der einfachen Geometrie des Objekts, das in diesem Fall nur aus zwei Flächen besteht, und der Tatsache, dass ausschließlich die Objektgeometrie zur Registrierung verwendet wurde.
Um diesem Problem zu begegnen, wurde ein komplexeres Objekt aufgenommen. Abbildung 6.33 zeigt das Registrierungsergebnis eines achtseitigen Volumens, dessen Punktwolken eine deutlich bessere Ausrichtung aufweisen. Zur weiteren Verbesserung wird für die folgenden Experimente ein Kopfmodell aus Kunststein verwendet (siehe Abbildung 6.34). Dieses Objekt verfügt über eine Geometrie mit vielen simplen Flächen in verschiedenen Ausrichtungen. Der Kunststein ist optimal für die Aufnahme geeignet, da die helle Farbe die Infrarotmuster gut reflektiert. Um das Registrierungsergebnis besser beurteilen zu können, wurden farbige Flächen auf der Oberfläche aufgebracht. In Abbildung 6.35 ist das Registrierungsergebnis des Modells dargestellt. Da sich die folgenden Ergebnisse visuell kaum unterscheiden, wird der RMS-Fehler als Bewertungsmetrik herangezogen, der bei der aktuellen Registrierung etwa 1,158 mm beträgt.

Abbildung 6.32: Paarweise lokale Registrierung eines Quaders

Abbildung 6.33: Paarweise lokale Registrierung eines achtseitigen Volumens

Um die Registrierung zu optimieren und die Stabilität zu verbessern, können die hinzugefügten farbigen Flächen genutzt werden. Open3D implementiert dazu eine Methode basierend auf [72]. Es wird eine neue Fehlermetrik eingeführt, die neben geometrischen auch farbliche Informationen berücksichtigt:

$$E(\mathbf{T}) = (1-\sigma) \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{K}} \left(r_C^{(\mathbf{p},\mathbf{q})}(\mathbf{T})\right)^2 + \sigma \sum_{(\mathbf{p},\mathbf{q})\in\mathcal{K}} \left(r_G^{(\mathbf{p},\mathbf{q})}(\mathbf{T})\right)^2 \tag{6.1}$$

Dabei steht $r_G^{(\mathbf{p},\mathbf{q})}(\mathbf{T})$ für den geometrischen Fehler, wie er bei der Point-to-Plane-Metrik genutzt wird, und $r_C^{(\mathbf{p},\mathbf{q})}(\mathbf{T})$ beschreibt den photometrischen Fehler, der die lokalen Gradienten der Farbintensitätswerte zwischen korrespondierenden Punkten erfasst. Der Gewichtungsfaktor σ ermöglicht die Anpassung des Verhältnisses zwischen den beiden Fehlertermen.
Abbildung 6.36 zeigt das Registrierungsergebnis mit dieser neuen Metrik. Der RMS-Fehler beträgt hierbei etwa 1,105 mm, was eine leichte Verbesserung darstellt. Da die

Abbildung 6.34: Testobjekt: Kopfmodell aus Kunststein

Abbildung 6.35: Paarweise lokale Registrierung des Kopfmodells

Transformation auf Basis der Farbunterschiede bestimmt wird, wurde eine weitere Aufnahme mit stärkeren Farbkontrasten durchgeführt. In Abbildung 6.37 ist das Ergebnis mit roten Flächen auf dem Objekt dargestellt. Der RMS-Fehler liegt konstant bei etwa 1,045 mm, sodass diese Metrik für die folgende Registrierung des Objekts verwendet wird.

Abbildung 6.36: Registrierung unter Berücksichtigung der Farbwerte

Abbildung 6.37: Registrierung mit erhöhtem Farbkontrast

6.12.2 Registrierung der Perspektiven

Nachdem die Punktwolkenpaare ausgerichtet wurden, erfolgt die Registrierung der Punktwolken aus verschiedenen Perspektiven. Ein naheliegender Ansatz ist die Verwendung desselben Registrierungsalgorithmus wie bei den Punktwolkenpaaren. Abbildung 6.38 zeigt die Punktwolken der Figur aus 16 Perspektiven. Das Ergebnis der Registrierung mit dem zuvor genutzten Verfahren ist in Abbildung 6.39 dargestellt. Dabei wurde mit der ersten Perspektive begonnen, die das Ursprungssystem definiert. Die zweite aufgenommene Punktwolke wurde zur ersten Aufnahme hin ausgerichtet. Das Ergebnis daraus wurde dann als Zielpunktwolke für die darauf folgende Perspektive genutzt.
Dieses sequentielle Vorgehen ist jedoch für die Registrierung mehrerer Perspektiven, wie

Abbildung 6.38: Zusammen dargestellte Punktwolken aller aufgenommenen Perspektiven auf das Objekt

Abbildung 6.39: Registrierte Punktwolken aller Perspektiven mit dem zuvor genutzten ICP-Verfahren

sie hier vorliegen ungeeignet. Während die ersten Perspektiven zuverlässig registriert werden, verschlechtern sich die Ergebnisse mit zunehmender Anzahl an Perspektiven. Dies liegt zum einen an der Akkumulation von Fehlern durch verrauschte Daten, die die Genauigkeit späterer Iterationen beeinträchtigen. Zum anderen ändert sich die Ausrichtung vorheriger Punktwolken, womit die initiale Ausrichtung der folgenden Punktwolken mehr und mehr hinfällig wird.

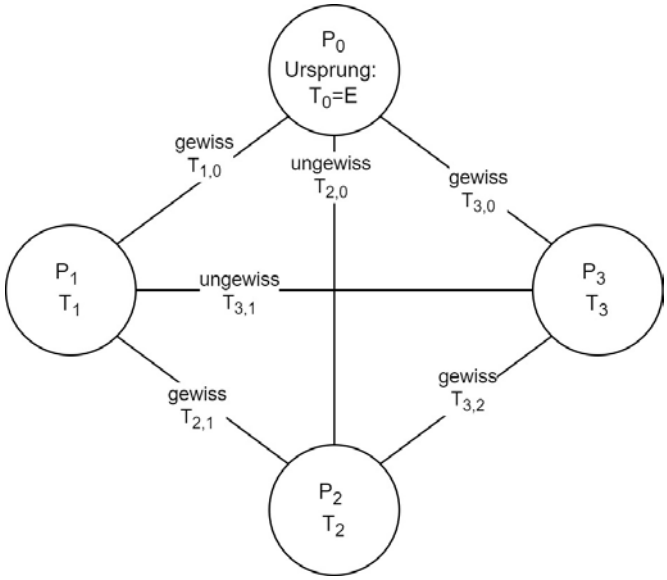


Abbildung 6.40: Pose-Graph mit vier Knoten und sechs Kanten

Um dieses Problem zu lösen, wird eine Methode nach [18] verwendet, die in Open3D implementiert ist. Diese Methode basiert auf einem *Pose-Graph*, einer Datenstruktur bestehend aus Knoten und Kanten. Knoten (P_i) repräsentieren die Punktwolken der verschiedenen Perspektiven und enthalten eine Transformationsmatrix (T_i), die die Punktwolke in das globale Koordinatensystem transformiert. Das globale Koordinatensystem wird durch die erste Punktwolke definiert, sodass T_0 der Einheitsmatrix entspricht. Die anderen Punktwolken befinden sich durch die Vorregistrierung mittels Markern bereits annähernd im globalen Koordinatensystem, wodurch auch deren Transformationsmatrix als Einheitsmatrix angenommen werden.
Die Knoten sind durch Kanten verbunden, die jeweils eine Transformationsmatrix ($T_{i,j}$) enthalten, welche die Ausgangspunktwolke P_i mit der Zielpunktwolke P_j in Übereinstimmung bringt. Benachbarte Punktwolken, die aufgrund ihrer Überlappung genau ausgerichtet werden können, sind mit Kanten verbunden, die als „gewiss“ gekennzeichnet sind. Die Transformationsmatrizen dieser Kanten werden mithilfe des zuvor verwendeten ICP-Verfahrens bestimmt. Kanten zwischen nicht benachbarten Punktwolken sind als „ungewiss“ markiert und verbinden Knoten mit potenzieller Überlappung. Jede Kante erhält eine Informationsmatrix, die die Unsicherheit der bestimmten Transformation beschreibt und als Gewichtung in der Optimierung dient. Abbildung 6.40 veranschaulicht einen solchen Pose-Graph.

Nach der Erstellung des Pose-Graphs wird eine Pose-Graph-Optimierung durchgeführt. Dabei werden die Punktwolken so angeordnet, dass eine global konsistente Gesamtpunktwolke entsteht. Alle den Kanten zugeordneten Transformationen werden genutzt, um die folgende Fehlerfunktion zu minimieren:

$$E(\mathbf{x}) = \sum_{(i,j)\in\mathcal{E}} \left(\mathbf{e}_{ij}^{\top} \Omega_{ij} \mathbf{e}_{ij} \right) \tag{6.2}$$

Dabei gilt:

- ε: Menge der Kanten im Pose-Graph.
- e_{ij}: Fehlervektor der Kante zwischen den Knoten i und j.
- Ω_{ij}: Informationsmatrix zur Gewichtung basierend auf der Unsicherheit der Transformation. Kanten mit höherer Zuverlässigkeit haben größeren Einfluss.

Der Fehlervektor ist definiert als:

$$\mathbf{e}_{ij} = \operatorname{Log}\left(\left(\mathbf{T}_{ij}^{\text{measured}} \right)^{-1} \left(\mathbf{T}_i^{-1} \mathbf{T}_j \right) \right) \tag{6.3}$$

Dabei gilt:

- T_i, T_j: Aktuelle Schätzungen der Transformationen der jeweiligen Punktwolken. In diesem Fall die Einheitsmatrix, da die Punktwolken vorregistriert sind.
- $T_{ij}^{measured}$: Durch lokale Registrierung ermittelte Transformation, die den Kanten zugeordnet ist.

Die Pose-Graph-Optimierung ist in Open3D gemäß [53] implementiert [66]. Die Ausrichtung der Punktwolken wird iterativ angepasst, um die Gesamtfehlerfunktion zu minimieren. Die Optimierung endet, wenn ein Konvergenzkriterium erreicht ist. Es werden die Standardparameter von Open3D verwendet [65].
Abbildung 6.41 zeigt das Ergebnis der Registrierung neben der vereinten Eingangspunktwolke. Obwohl der Unterschied visuell schwer zu erkennen ist, deuten die Farben der einzelnen Perspektiven darauf hin, dass die Punkte in den Flächen des Modells

(a) Vereinte Punktwolke vor der Registrierung

(b) Vereinte Punktwolke nach der Registrierung

Abbildung 6.41: Ergebnis der Pose-Graph-Optimierung

stärker verschmelzen. Während der Optimierung konvergiert der Residualwert, der dem Wert der Gesamtfehlerfunktion entspricht, schnell gegen eins, was darauf hindeutet, dass die Ausrichtungsfehler so weit wie möglich minimiert wurden.

Mit dieser Erkenntnis wird die Registrierung abgeschlossen. Wie genau die finale Punktwolke das Objekt repräsentiert, wird im folgenden Kapitel, beim Testen des Systems, überprüft.

6.12.3 Implementierung (Registrator)

Die beschriebenen Methoden zur Registrierung der Punktwolken sind in der Klasse *Registrator* implementiert. Das Klassendiagramm dieser Klasse ist in Abbildung 6.42 dargestellt.

Konstruktor

Beim Erstellen einer Instanz wird ein Objekt vom Typ *DatasetManager* übergeben, das als Attribut des Registrator-Objekts gespeichert wird. Dieses Attribut enthält alle zu verarbeitenden Punktwolken sowie weitere Dateneigenschaften wie die mittleren Punktabstände der jeweiligen Datensätze.

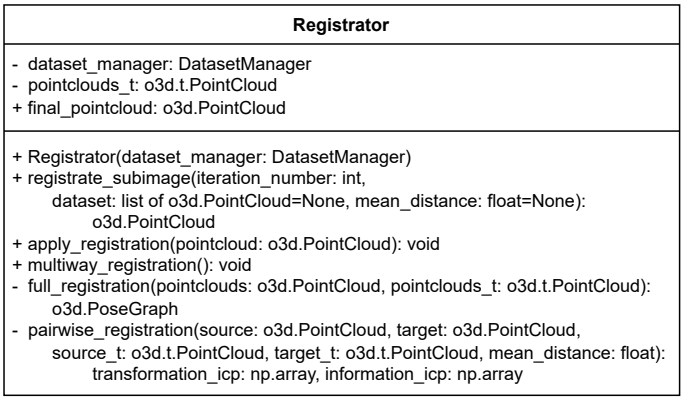


Abbildung 6.42: Klassendiagramm der Klasse *Registrator*

Paarweise Registrierung

Die paarweise Registrierung erfolgt in der Klassenmethode *registrate_subimage*. Diese Methode wird im Prozessablauf automatisch nach der Filterung aufgerufen, wobei die ID des zu registrierenden Datensatzes übergeben wird. Basierend auf den mittleren Punktabständen des aktuellen Datensatzes wird ein Parametersatz erstellt, mit dem die multiskalare Registrierung durchgeführt wird. Die Funktion gibt die zusammengefügte Punktwolke zurück. Diese kann vom Benutzer über das GUI bestätigt oder verworfen werden, woraufhin eine neue Aufnahme der Perspektive erfolgt. Wird der Datensatz bestätigt, wird die Methode *apply_registration* aufgerufen, die die Punktwolke dem Datensatz der registrierten Punktwolkenpaare hinzufügt.

Registrierung der Perspektiven

Die Hauptmethode für die Registrierung der Perspektiven ist *multiway_registration*. Nach ihrem Aufruf wird zunächst der Pose-Graph aus allen Punktwolken erstellt, indem die Methode *full_registration* aufgerufen wird. Diese bestimmt mit *pairwise_registration* die Transformationsanweisungen der Kanten benachbarter Punktwolken. Nach der Erstellung wird der Pose-Graph optimiert, und die finale Punktwolke entsteht durch Ausrichtung der einzelnen Perspektiven mit den ermittelten Transformationen.

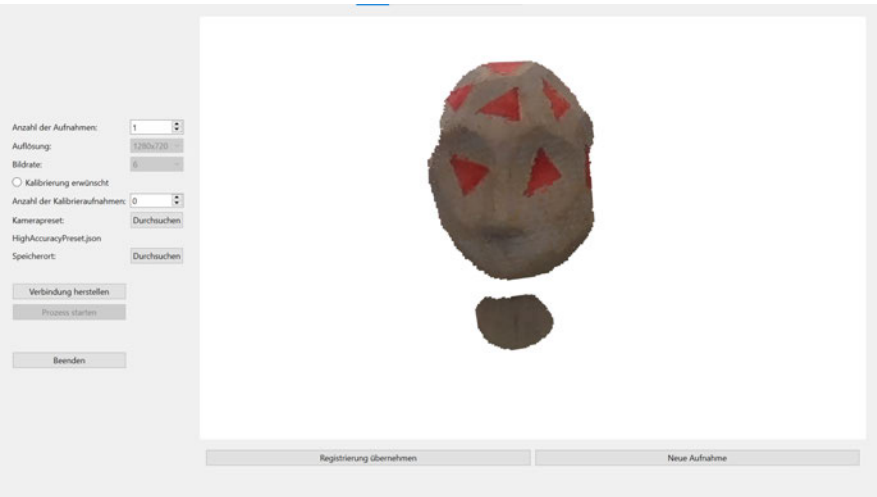


Abbildung 6.43: GUI zur Steuerung der Registrierung

6.12.4 Änderungen im UI

Nach Abschluss der Filterung der Punktwolken wird die Filterungsansicht durch eine Benutzeroberfläche für die paarweise Registrierung ersetzt. Zwei Schaltflächen werden hinzugefügt: *Registrierung übernehmen* und *Neue Aufnahme*. Für jede Perspektive wird zunächst die Registrierung durchgeführt und anschließend das Ergebnis angezeigt. Entspricht die Registrierung den Erwartungen, kann sie übernommen werden; andernfalls kann eine neue Aufnahme gestartet werden. Falls die Registrierung fehlschlägt, wird automatisch eine neue Aufnahme initiiert. Die aktualisierte GUI ist in Abbildung 6.43 dargestellt.
Nach Abschluss der paarweisen Registrierung wird automatisch die globale Registrierung aller Perspektiven durchgeführt, die keine weitere Benutzerinteraktion erfordert. Sobald dieser Vorgang abgeschlossen ist, wird das Ergebnis in einem neuen Fenster visualisiert. Durch Schließen dieses Fensters wird der Prozess fortgesetzt und der nächste Schritt der Verarbeitungskette eingeleitet.

6.13 Oberflächengenerierung und Datenexport

An dieser Stelle werden die vorbereiteten Daten zu einer zusammenhängenden Oberfläche zusammengefügt und anschließend in einem geeigneten Datenformat exportiert. Zunächst werden verschiedene Methoden der Oberflächengenerierung beschrieben und

die in dieser Arbeit gewählte Lösung vorgestellt. Anschließend werden die Implementierung dieser Methode und der Datenexport erläutert.

6.13.1 Oberflächengenerierung

Der Prozess der Oberflächengenerierung besteht darin, aus den ermittelten Punkten zusammenhängende Dreiecke zu bilden. Diese benachbarten Dreiecke formen die Oberfläche des resultierenden Modells. Die Bibliothek Open3D implementiert drei verschiedene Methoden zur Oberflächengenerierung, die im Folgenden beschrieben werden.

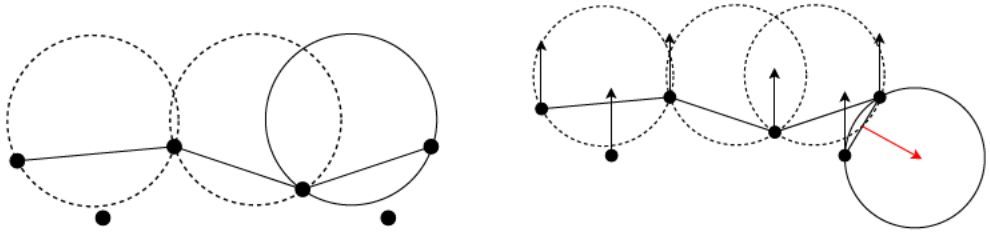
Alpha Shapes

Die Alpha-Shape-Methode, ursprünglich in [23] veröffentlicht, ermöglicht die Definition und Bestimmung der Form einer Menge von Punkten im zwei- oder dreidimensionalen Raum. Im Folgenden wird dieses Verfahren basierend auf [31] erläutert.
Ein häufig verwendeter Ansatz zur intuitiven Erklärung des Verfahrens nutzt die Analogie eines Eisbechers. Die Punkte der Punktwolke entsprechen dabei festen Bestandteilen, wie Schokoladenstückchen im Eis, während der Raum um die Punkte der Eiscreme selbst entspricht. Mit einem kugelförmigen Löffel wird überall dort das Eis entfernt, wo es erreichbar ist; die Punkte fungieren als Hindernisse, die nicht entfernt werden können. Auf diese Weise wird das Eis so weit wie möglich entfernt. Das namensgebende Parameter α entspricht dabei dem Radius des kugelförmigen Löffels. Das nach dem Entfernen übrig gebliebene Objekt wird als *Alpha Shape* bezeichnet, wobei die entstehenden runden Kanten begradigt werden.
Gemäß dieser Analogie führt ein gegen Null gehendes α dazu, dass das resultierende Alpha Shape den ursprünglichen Punkten entspricht. Umgekehrt nähert sich das Alpha Shape bei α gegen Unendlich der konvexen Hülle der Punkte an, da kein innenliegendes Volumen entfernt wird. Somit ist die Detailtiefe der erzeugten Oberfläche durch die Wahl von α einstellbar, wobei der Wert größer sein sollte als der durchschnittliche Abstand zwischen den Punkten.
Bei der Auswahl des α-Parameters gibt es jedoch Einschränkungen. Die optimale Wahl hängt stark von den verwendeten Daten ab. Bei einer einfachen Oberflächenstruktur kann ein großer α-Wert die Oberfläche rekonstruieren. Verfügt das aufgenommene Objekt jedoch über Aussparungen oder feine Details, können diese möglicherweise nur mit

einem kleineren α-Wert rekonstruiert werden, der unter einem bestimmten Schwellenwert liegt. Abhängig von diesem Wert kann das rekonstruierte Objekt in Bereichen mit kleinen Radien fehlerhaft sein. Selbst wenn ein geeigneter Wert gefunden wurde, kann dieser bei ungleichmäßiger Abtastung - wie sie in der vorliegenden Arbeit auftritt (bei Aufnahmen von Flächen, die sich parallel zur Sensorfläche befinden, und anderen, die eher orthogonal zur Sensorfläche stehen) - in verschiedenen Bereichen der Punktwolke nicht mehr optimal passen.
Zudem ist die Methode in der Implementierung von Open3D darauf ausgelegt, dass die Oberfläche mit einem einzigen α-Wert erzeugt wird. Daher muss dieser Wert sorgfältig gewählt werden, um ein möglichst gutes Gesamtergebnis zu erzielen.

Ball-Pivoting-Algorithmus

Der BPA wurde als speichereffizienter und schnell ausführbarer Algorithmus entwickelt, um aus Punktwolken Dreiecksnetze (Meshes) zu erzeugen [10]. Der Algorithmus basiert auf der Idee, eine Kugel über die Punktwolke zu rollen, um die Oberfläche zu rekonstruieren. Er beginnt mit drei Punkten, die ein sogenanntes *Seed-Dreieck* bilden. Diese werden so ausgewählt, dass die Kugel diese drei Punkte berührt, dabei aber keine weiteren Punkte enthält. Anschließend wird die Kugel um eine Kante des Dreiecks rotiert. Wenn die Kugel dabei einen weiteren Punkt berührt, wird das neu gebildete Dreieck dem Mesh hinzugefügt. Jede Kante jedes neu hinzugefügten Dreiecks wird auf mögliche weitere Dreiecke überprüft. Kanten, bei denen kein weiterer Punkt gefunden wird, werden als Grenzkanten markiert.
Da die Kugel bildlich gesprochen über die Oberfläche rollt - vorausgesetzt, die Punktwolke ist dicht genug - werden Ausreißer, die sich unterhalb dieser Fläche befinden, beim Erzeugen der Oberfläche nicht berücksichtigt. Somit kann abhängig von der Größe der Kugel die Oberfläche in gewissem Maße gefiltert werden. In Abbildung 6.44a ist dieses Verhalten anhand eines Beispiels im zweidimensionalen Raum dargestellt. Anstatt von Dreiecken werden dabei Linien gebildet, die die Punkte verbinden; zudem wird um einen Punkt und nicht um eine Kante rotiert.
Während der Durchführung werden weitere Kriterien genutzt, um zu überprüfen, ob ein neu gefundenes Dreieck gültig ist. Beispielsweise werden die Flächennormalen in den betrachteten Bereichen herangezogen. Für jedes gebildete Dreieck wird überprüft, ob das Skalarprodukt des Normalenvektors des neuen Dreiecks mit dem Normalenvektor der zugrundeliegenden Oberfläche in diesem Bereich innerhalb eines bestimmten



(a) Ausschließen von Ausreißern (b) Überprüfen der Normalenvektoren

Abbildung 6.44: Visualisierung des BPA

Wertebereichs liegt. In Abbildung 6.44b ist dieser Vorgang dargestellt. Die schwarzen Pfeile repräsentieren die Normalenvektoren der zugrundeliegenden Oberfläche, während der rote Pfeil den Normalenvektor des neu gebildeten Dreiecks bzw. der Linie darstellt. Auch schlecht abgetastete Bereiche können verarbeitet werden. Wenn die Punktdichte in einem Bereich so gering ist, dass die Kugel hindurchfallen würde, wird die entsprechende Kante zunächst als Grenze markiert. Anschließend kann der Algorithmus mit einer größeren Kugel erneut durchlaufen werden, um die noch vorhandenen Löcher zu schließen.
Im Gegensatz dazu kann es in Bereichen mit sehr feinen Details vorkommen, dass diese von der Kugel nicht erfasst werden können. Dieser Fall kann vom Algorithmus nicht abgefangen werden und resultiert in einem Informationsverlust. Daher muss die Größe der verwendeten Kugel an die erwartete Größe der Details angepasst werden. So können zunächst mit kleinen Kugeln dichte, detailreiche Bereiche erfasst und anschließend Bereiche mit geringerer Punktdichte mit größeren Kugeln rekonstruiert werden.

Poisson-Flächenrekonstruktion

Die Poisson-Flächenrekonstruktion unterscheidet sich von den beiden vorherigen Methoden dadurch, dass sie eine glatte Oberfläche erzeugt, die nicht zwingend exakt durch die aufgenommenen Punkte verläuft. Stattdessen wird eine mathematische Funktion approximiert, die den Raum in Bereiche innerhalb und außerhalb der Objektoberfläche unterteilt. Dadurch kann die Oberfläche rekonstruiert werden [50]. Im Folgenden wird das Prinzip der Rekonstruktion nach [60] kurz beschrieben:

- Die Normalenvektoren für jeden Punkt der Eingangspunktwolke werden bestimmt und bilden zusammen ein Vektorfeld $\mathbf{V}$.

- Eine Indikatorfunktion $\mathbf{X}(\mathbf{p})$, die den dreidimensionalen Raum beschreibt, wird definiert, sodass bei einen Punkt $\mathbf{p}$ gilt:

$$\mathbf{X}(\mathbf{p}) = \begin{cases} 1, & \mathbf{p} \in \mathbf{M} \\ 0, & \mathbf{p} \notin \mathbf{M} \end{cases} \tag{6.4}$$

  wobei $\mathbf{M}$ die Objektoberfläche darstellt.

- Der Gradient der Indikatorfunktion ∇X ist nur in der Nähe der Oberfläche ungleich Null.

- Es wird angenommen, dass der Gradient ∇X dem Vektorfeld der Normalen $\mathbf{V}$ entspricht:

$$\nabla X = \mathbf{V} \tag{6.5}$$

- Dies führt zu einer Poisson-Gleichung, die numerisch gelöst werden kann, um die Funktion $\mathbf{X}$ zu bestimmen. Dabei wird die Indikatorfunktion an das Vektorfeld der Normalen angepasst.

- Abschließend wird mit dem Marching-Cubes-Algorithmus [57] die Oberfläche als Isofläche der Funktion $\mathbf{X}$ extrahiert und ausgegeben.

Der Vorgang wird durch die Verwendung von Octrees unterstützt. Ein Octree ist, ähnlich wie ein K-d-Baum, eine Datenstruktur zur effizienten Verwaltung mehrdimensionaler Daten. Während ein Bereich beim K-d-Baum in zwei kleinere Bereiche unterteilt wird, wird der Raum beim Octree in bis zu acht Teilbereiche unterteilt.

Ergebnisse

An dieser Stelle werden die drei Methoden miteinander verglichen, um den am besten geeigneten Ansatz zu identifizieren. Für die Bewertung sind die Genauigkeit bzw. die Güte der Rekonstruktion sowie die Verarbeitungsgeschwindigkeit relevant. Da die Genauigkeit quantitativ schwer zu bewerten ist, wird in diesem Punkt auf eine visuelle und somit eher subjektive Bewertung zurückgegriffen. Um die Anforderungen hinsichtlich der zeitlichen Beschränkung (**SW-GE 5**) erfüllen zu können, wird auch die Verarbeitungszeit berücksichtigt.

Abbildung 6.45: Zusammengefügte Punktwolke der Figur

Die Rekonstruktionen wurden alle auf einem zuvor aufgenommenen Datensatz aus 48 Perspektiven durchgeführt, der die zuvor verwendete Figur darstellt. Aufgrund der Objektgeometrie enthält die Gesamtpunktwolke einen Bereich, in dem keine Punkte aufgenommen werden konnten bzw. die herausgefiltert wurden. Dieser Umstand führt zu einer nicht optimalen Eingangspunktwolke, spiegelt jedoch die Realität recht gut wider. Bei einer so großen Anzahl an Aufnahmen kann es durchaus passieren, dass einzelne Punktwolken von minderer Qualität sind und Teile des Objekts unterrepräsentiert oder gar nicht vorhanden sind. Die Gesamtpunktwolke wurde erstellt, indem die aufgenommenen Perspektiven anhand der zuvor beschriebenen Methoden verarbeitet und registriert wurden. In Abbildung 6.45 ist die Punktwolke dargestellt, bevor sie von den Rekonstruktionsmethoden verarbeitet wird. Gut zu erkennen ist der Bereich im Halsbereich der Statue, der über keine Punkte verfügt.

Alpha Shapes: Bei dieser Methode wird die Detailgenauigkeit über den α-Parameter bestimmt. Dieser wird in den Tests als ein Vielfaches des mittleren Punktabstands ($\overline{pa}$) festgelegt. In Abbildung 6.46a ist die rekonstruierte Punktwolke dargestellt, bei der der α-Parameter in etwa dem mittleren Punktabstand der Eingangspunktwolke entspricht. Die Detailtreue ist in diesem Fall maximal. Da quasi jeder Punkt zur Generierung der

Oberfläche genutzt wird, entsteht jedoch eine stark zerklüftete Oberflächenstruktur. Der Bereich, in dem keine Punkte vorhanden waren, wird durch die verhältnismäßig geringe Punktdichte in diesem Bereich noch vergrößert und nicht aufgefüllt. Aufgrund der nicht optimalen Qualität der erzeugten Punktwolken führt dies zu einer unschönen Oberfläche.
In Abbildung 6.46b ist die Oberfläche dargestellt, die mit einem α-Parameter erzeugt wurde, der etwa dem Siebenfachen des mittleren Punktabstands entspricht. Die Oberfläche verfügt noch über eine gute Detailtreue, wobei der Bereich im Halsbereich weiterhin nicht aufgefüllt werden konnte.
Als letzte Vergleichsrekonstruktion ist in Abbildung 6.46c die Oberfläche dargestellt, die mit einem α-Parameter generiert wurde, der dem 22-fachen des mittleren Punktabstands entspricht. Dieser Wert wurde so gewählt, dass die Oberfläche geschlossen werden kann. Dabei ist jedoch zu erkennen, dass die Details zunehmend schlechter abgebildet werden. Zum Beispiel wurden mehrere kleine Dreiecke zu größeren Dreiecken zusammengefasst, wodurch einige Details verloren gehen. Dies ist gut im Bereich der Figur zu erkennen, wo sich der Mund befinden würde. Zwar wurde mit diesem Parameter eine wasserdichte Oberfläche erstellt, allerdings sind die gebildeten Dreiecke im Halsbereich direkt mit den beteiligten Punkten verbunden, ohne jegliche Approximation. Dadurch entspricht das Modell noch weniger der Realität. Dieses Phänomen lässt sich auf die gesamte Oberfläche anwenden. Da die erzeugte Oberfläche ausschließlich auf den verfügbaren Punkten basiert und diese durch Dreiecke direkt verbunden sind, werden keine natürlichen, approximierten Kurven und Krümmungen zwischen den Punkten erzeugt. Bei einer Punktwolke mit einer extrem hohen Punktauflösung führt dies zu einer natürlich aussehenden Oberfläche. In diesem Fall jedoch besteht die Oberfläche aus sichtbaren Dreiecken, und Bereiche wie der Hals bestehen aus flachen Ebenen. Die Erstellung der Oberflächen aus den 48 Perspektiven dauert bei jedem α-Parameter etwa 50 Sekunden.

Ball-Pivoting-Algorithmus: Wie zuvor beschrieben wird die Detailtiefe beim BPA durch die übergebenen Radien gesteuert. Im Rahmen der Umsetzung wurde festgestellt, dass die Verarbeitungsdauer mit den vorhandenen Punktwolken zu lang ist. Der Prozess war mit dem genutzten Computer auch nach 48 Minuten nicht abgeschlossen, was gemäß den Anforderungen die maximal zulässige Verarbeitungsdauer bei der gegebenen Anzahl an aufgenommenen Perspektiven darstellt. Um den Vorgang innerhalb der zulässigen Zeit abschließen zu können, musste die Punktwolke etwa um den Faktor 2 herunters-

(a) $\alpha \approx \overline{pa}$ (b) $\alpha \approx 7\overline{pa}$ (c) $\alpha \approx 22\overline{pa}$

Abbildung 6.46: Ergebnisse der Rekonstruktion bei verschiedenen α-Parametern

kaliert werden. Dabei werden die Punkte in einem gewissen Bereich zu einem Punkt zusammengefasst, was mit einem Verlust von Details einhergeht. Damit der Vorgang durchgeführt werden kann, müssen die übergebenen Radien an die neuen Punktabstände angepasst werden. Diese werden aufsteigend nach der Größe übergeben. Wie schon bei der vorherigen Methode werden diese anhand des mittleren Punktabstands ($\overline{pa}$) definiert. Da dieser durch das Herunterskalieren bekannt ist, wird ein Vielfaches dieses Wertes verwendet. Der Punktabstand der Rohdaten liegt bei etwa 0,7 mm. Nach dem Herunterskalieren liegen die Punkte in einem Raster von 1,3 mm. Der kleinste Radius wird als das Doppelte des Punktabstands, also 3 mm, angenommen. Für die weiteren Radien werden jeweils die nächsten halben und vollen Zehnerpotenzen angenommen, bis zu einem maximalen Radius von 10 cm.
Damit der Vorgang durchgeführt werden kann, werden die Normalenvektoren benötigt, die zuvor bestimmt werden müssen. Für die Bewertung der zeitlichen Leistung wird dieser Vorgang daher ebenfalls betrachtet.
In Abbildung 6.47 ist die mit dem Algorithmus erzeugte Oberfläche dargestellt. Diese ist durch die Reduktion der Punktdichte weniger detailreich. Zudem weist die Oberfläche kleine Berge und Täler auf, die aus den Punkten resultieren, die sich an der Oberfläche der Punktwolke befinden und durch Ungenauigkeiten in der Punktwolke eine Struktur aufweisen, die nicht dem realen Objekt entsprechen. Das Loch in der Punktwolke konnte durch den Algorithmus auch mit größeren Radien nicht geschlossen werden. Die Verarbeitungsdauer liegt bei den eingestellten Parametern bei etwa 36 Sekunden. Wie bei

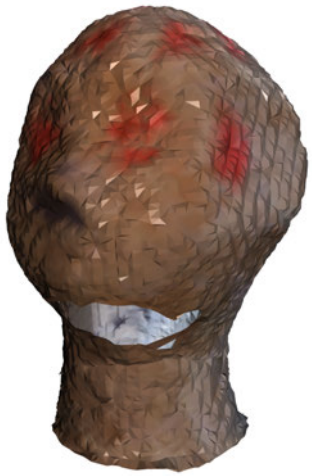


Abbildung 6.47: Rekonstruierte Oberfläche mit Hilfe des Ball-Pivoting-Algorithmus

der Alpha-Shape-Methode verläuft die erstellte Oberfläche exakt durch die verwendeten Punkte. Damit werden die Teilflächen in keiner Form approximiert, um eine ästhetisch ansprechendere Abbildung zu erzeugen.

Poisson-Flächenrekonstruktion: Der Detailgrad der rekonstruierten Fläche hängt bei dieser Methode vor allem von der Tiefe d des verwendeten Octrees ab. Mit diesem Parameter wird die Auflösung des resultierenden Meshes bestimmt. Für den Test der Methode wird die Tiefe variiert, um bei den gegebenen Punktwolkendaten die bestmögliche Oberfläche zu erzeugen. Bei dieser Methode wird eine geschlossene Oberfläche erzeugt, die am besten zu den übergebenen Punkten passt. Dadurch entsteht eine durchgehende Oberfläche, die in Bereichen, die schlecht oder gar nicht aufgenommen werden konnten, approximiert wird.
In Abbildung 6.48a ist die rekonstruierte Oberfläche bei einer Octree-Tiefe von 8 dargestellt. In Abbildung 6.48b ist die Oberfläche mit einer Octree-Tiefe von 9 erzeugt worden. Daneben ist in Abbildung 6.48c das Ergebnis der Rekonstruktion bei einer Octree-Tiefe von 10 dargestellt.
Bei einer Tiefe von 8 erscheint die rekonstruierte Oberfläche wie durch einen Filter geglättet. Kleinere Variationen in den Punkten werden scheinbar ignoriert, was zu einer glatteren, detailärmeren Oberfläche führt. Mit diesem Parameter wird die Oberfläche zu stark vereinfacht.
Bei einer Tiefe von 10 weist die Oberfläche eine sehr raue Struktur auf. Durch die sehr feine Diskretisierung des Raumes werden kleinste Änderungen der Punkte im Ergebnis

(a) $d = 8$ (b) $d = 9$ (c) $d = 10$

Abbildung 6.48: Ergebnisse der Rekonstruktion bei verschiedenen d-Parametern

sichtbar. Damit werden die Punkte nicht ausreichend gemittelt, wodurch die Oberfläche die verrauschte Punktstruktur widerspiegelt.
Die Rekonstruktion bei einer Tiefe von 9 scheint für diesen Datensatz optimal geeignet. Die Oberfläche ist ausreichend geglättet, sodass einzelne Ungenauigkeiten in der Punktwolke nicht zu erkennen sind. Dennoch sind die Details des Modells erhalten und werden nicht herausgefiltert. Je nach verwendeter Tiefe liegen die Ergebnisse dieser Methode in etwa 194 bis 239 Sekunden vor.

Entscheidung

Die Poisson-Flächenrekonstruktion erweist sich für die vorliegende Anwendung als am besten geeignet. Eventuell nicht abgetastete Bereiche werden zuverlässig geschlossen, was bei den verwendeten Daten ein wichtiger Aspekt ist. Zudem bleibt die Struktur, die durch die Punktwolke vorgegeben ist, (bei einem gut gewählten Tiefenparameter) weitestgehend erhalten und wird nur geringfügig geglättet bzw. vereinfacht. Der Tiefenparameter sollte abhängig von den eingegebenen Punktwolken eingestellt werden. Bei größeren Objekten sollte eine feinere Diskretisierung verwendet werden als bei kleineren Objekten und somit kleineren Punktwolken.

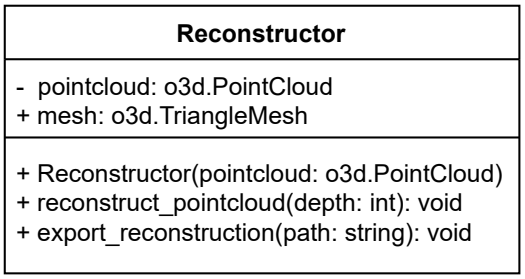


Abbildung 6.49: Klassendiagramm der Klasse *Reconstructor*

6.13.2 Implementierung (Reconstructor)

Die Methoden zur Rekonstruktion und zum Export der Daten sind in der Klasse *Reconstructor* implementiert. Das Klassendiagramm dieser Klasse ist in Abbildung 6.49 dargestellt.

Konstruktor

Beim Erstellen einer Instanz wird die zuvor erstellte Gesamtpunktwolke übergeben und gespeichert. Anschließend werden die Normalenvektoren der Punktwolkenoberfläche ermittelt und so ausgerichtet, dass sie eine konsistente Orientierung aufweisen und nicht manche nach innen und andere nach außen zeigen.

Durchführung der Rekonstruktion

Die eigentliche Rekonstruktion der Oberfläche wird in der Methode *reconstruct_ pointcloud* durchgeführt. Dabei wird die vom Benutzer eingestellte Tiefe des Octrees als Parameter übergeben. Die Methode gibt keinen Wert zurück.

Export der Rekonstruktion

Nachdem die Rekonstruktion durchgeführt, angezeigt und vom Benutzer bestätigt wurde, wird die erzeugte Oberfläche exportiert. Abhängig von der eingestellten Dateiendung wird das Modell entweder im *stl*- oder im *obj*-Format gespeichert. Damit werden die

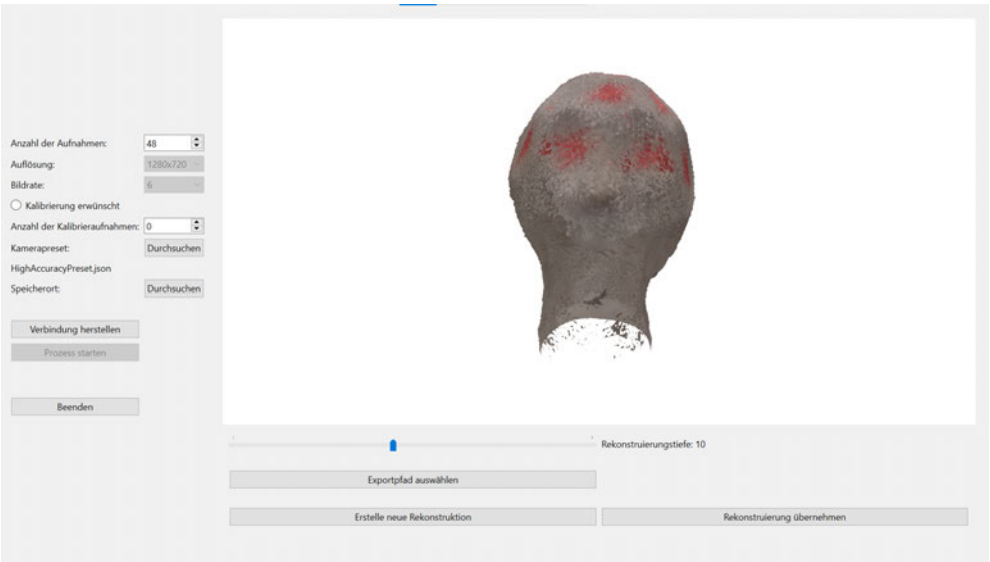


Abbildung 6.50: Benutzerschnittstelle Rekonstruktion

Anforderungen **SW-DV 1** und **SW-DV 2** erfüllt, die vorsehen, dass die Daten je nach Anwendungsfall in einem der beiden Formate ausgegeben werden.

6.13.3 Änderungen im UI

Zur Steuerung der Rekonstruktion wird dem Benutzer eine Oberfläche mit den folgenden Eingabemöglichkeiten präsentiert. Die Tiefe des Octrees wird über einen Schieberegler eingestellt. Nach Anpassung des Schiebereglers kann mit der Schaltfläche *Erstelle neue Rekonstruktion* die aktuell angezeigte Rekonstruktion verworfen und eine neue erstellt sowie angezeigt werden. Ist der Benutzer mit dem Detailgrad der Rekonstruktion zufrieden, kann er diese mit der Schaltfläche *Rekonstruktion übernehmen* speichern. Um einen Speicherpfad und das gewünschte Ausgabeformat festzulegen, kann mit der Schaltfläche *Exportpfad auswählen* ein Dateimanagerdialog geöffnet werden, in dem der Pfad zum gewünschten Speicherort und das Dateiformat eingestellt werden können. Über den Eingabemöglichkeiten werden wie zuvor die aktuellen Daten, in diesem Fall die aktuelle Rekonstruktion, dargestellt. In Abbildung 6.50 ist die geänderte grafische Benutzeroberfläche (GUI) dargestellt.

6.14 Zusammenfassung der Softwarekomponenten

In diesem Abschnitt wird abschließend zur Umsetzung das Softwaresystem in seiner Gesamtheit dargestellt. Das zentrale Element des Prozessablaufs bildet die Klasse *MainWindow*, in der das GUI implementiert ist. Zudem wird in dieser Klasse der Programmablauf anhand der zuvor beschriebenen Ablaufdiagramme mittels Signalen und Slots realisiert.
Abbildung 6.51 zeigt das Klassendiagramm der Klasse *MainWindow* sowie das Zusammenspiel mit den anderen Klassen. Aus Gründen der Übersichtlichkeit sind die Klassen der UI-Elemente nicht dargestellt, da sie für den grundlegenden Programmablauf keine wesentliche Rolle spielen. Außerdem werden die bereits beschriebenen Klassen ohne ihre Methoden und Variablen dargestellt. Ein vollständiges Klassendiagramm mit allen Methoden und Variablen der Klassen befindet sich im Anhang.

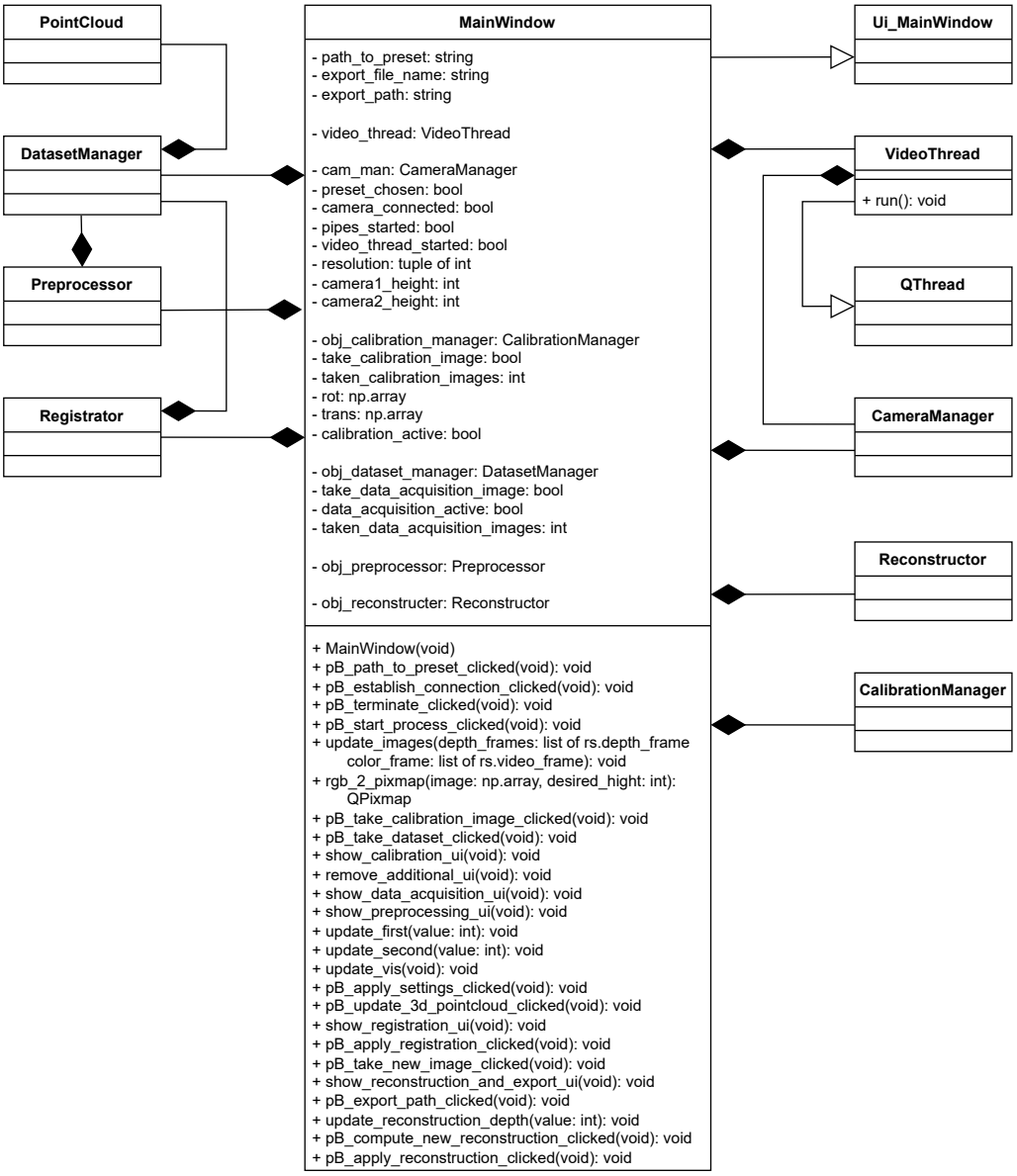


Abbildung 6.51: Klassendiagramm der Klasse *MainWindow* und Zusammenspiel aller Klassen

7 Test

Nachdem im vorherigen Kapitel die Umsetzung des Projekts detailliert beschrieben wurde, erfolgt in diesem Kapitel die Evaluation und Validierung der Ergebnisse. Dabei wird überprüft, ob die Anforderungen aus Kapitel 4 erfüllt wurden. Die Überprüfung der Anforderungen erfolgt nach dem folgenden Schema gemäß [48]:

- Testgegenstand: Die aktuell zu prüfende Anforderung.
- Durchführung: Beschreibung des Testvorgangs.
- Akzeptanzkriterium: Der quantitativ messbare Parameter, dessen Erfüllung den Test als bestanden kennzeichnet.
- Beobachtung: Die Ergebnisse des Tests.
- Beurteilung: Aussage über das Bestehen des Tests.

Die Erfüllung der Anforderungen wird in drei Abstufungen bewertet. Eine Anforderung gilt als **erfüllt**, wenn alle Testparameter innerhalb der geforderten Toleranzen liegen. Sie gilt als **teilweise erfüllt**, wenn nicht alle Kriterien erfüllt sind oder diese nicht im erwarteten Bereich liegen. Wenn keines der Kriterien erfüllt ist, gilt die Anforderung als **nicht erfüllt**.

7.1 Hardwareanforderungen

In diesem Abschnitt werden die Hardwareanforderungen überprüft. Dabei erfolgt die Prüfung entsprechend der in Kapitel 4 beschriebenen Unterteilung in allgemeine Anforderungen, Anforderungen an die Sensoreinheit sowie Anforderungen an den Aufnahmebereich.

7.1.1 Generelle Anforderungen

HW-GE 1 (NF)

Testgegenstand: Die Kosten für die Neuanschaffung von Hardwarekomponenten dürfen 250 € nicht überschreiten.
Durchführung: Die im Rahmen des Projekts anfallenden Kosten werden aufgelistet und addiert, um festzustellen, ob sie unterhalb des vorgegebenen Grenzwerts liegen.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn die Gesamtkosten den Wert von 250 € nicht überschreiten.
Beobachtung: Die Gesamtkosten für die beschafften Komponenten setzen sich wie folgt zusammen:

- Kamerahalterungen und Griffstück: 1,17 €
- Hartschaumplatten für den Aufnahmebereich: 55,36 €
- Hartschaumplatten für ArUco-Marker: 17,84 €

Sonstige Hardware, wie etwa Synchronisationskabel für die Kameras, waren bereits vorhanden und werden daher nicht in die Kostenkalkulation einbezogen. Die Gesamtkosten betragen somit 74,37 €.
Beurteilung: Die vorgegebene Kostengrenze von 250 € wird deutlich unterschritten. Damit ist diese Anforderung **vollständig erfüllt**.

HW-GE 2 (NF)

Testgegenstand: Das System muss während des Betriebs sicher sein und darf keine Gefahr für Personen darstellen. Potenzielle Gefahren sind zu identifizieren und gemäß geltender Normen und Vorschriften zu minimieren.
Durchführung: Der übliche Aufbau bei einem Scan mit dem vorliegenden System wird analysiert. Dabei werden mögliche Gefahrenquellen identifiziert und gegebenenfalls vorhandene Maßnahmen zu deren Minimierung oder Verhinderung bewertet.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn keine Gefahrenquelle existiert, für die keine geeigneten Maßnahmen zur Vermeidung oder Minimierung vorliegen.
Beobachtung: Das aktuelle System besteht aus Hartschaumplatten (für Kalibrierobjekt, Aufnahmebereich und ArUco-Marker), der Sensoreinheit mit Anschlusskabeln sowie einem Laptop, auf dem die Software ausgeführt wird.

- **Hartschaumplatten:**
  Nach den *Technischen Regeln für Arbeitsstätten* (Arbeitsstättenregeln - ASR) [7], herausgegeben von der Bundesanstalt für Arbeitsschutz und Arbeitsmedizin, gelten bereits Höhenunterschiede ab 4 mm als potenzielle Stolperstellen. Die verwendeten Hartschaumplatten weisen eine Dicke von 5 mm auf. Zwar könnte ein abgewinkelter Trittschutz eingesetzt werden, um die Stolpergefahr zu reduzieren, jedoch würde eine derartige Modifikation die Bedingungen für die Datenaufnahme eventuell beeinträchtigen. Da die Platten zudem bei punktueller Belastung relativ druckempfindlich sind, wird auf solche Maßnahmen verzichtet und der Benutzer angewiesen, den Aufnahmebereich nicht zu betreten und besondere Vorsicht walten zu lassen.

- **Sensoreinheit:** Die verwendeten Kameras sind mit Infrarotprojektoren ausgestattet, die gemäß Herstellerangaben und Normen der Laserklasse 1 angehören und somit als sicher eingestuft werden. Es besteht demnach kein Risiko für den Benutzer durch die Infrarotstrahlung. Auch mechanische Gefahren oder elektrische Risiken werden durch sichere Kabelverbindungen und den Einsatz von Niederspannungstechnik (Sichergestellt durch die USB-Schnittstelle) vermieden.

- **Anschlusskabel:** Die zum Einsatz kommenden Kabel sind so kurz dimensioniert, dass der Laptop in unmittelbarer Nähe zur Sensoreinheit positioniert werden muss. Da die Sensoreinheit in der Hand gehalten wird, ist eine Stolpergefahr durch Kabel weitestgehend ausgeschlossen. Somit entsteht auch durch die Kabel keine relevante Gefahrenquelle.

Beurteilung: Obwohl alle relevanten Gefahrenquellen analysiert und soweit möglich minimiert wurden, verbleibt eine gewisse Stolpergefahr durch die Hartschaumplatten. Unter diesem Aspekt gilt die Anforderung als **teilweise erfüllt**.

7.1.2 Anforderungen Sensoreinheit

HW-SE 1 (F)

Testgegenstand: Die Sensoreinheit erfasst die Oberflächengeometrie des Objekts.
Durchführung: Im Verlauf des Prozesses nimmt die Sensoreinheit notwendigerweise Daten auf, um eine Punktwolke zu erzeugen. Zur Überprüfung wird der Prozess normal ausgeführt.

Akzeptanzkriterium: Sobald aus den aufgenommenen Daten eine Punktwolke entsteht, in der die Form des Objekts erkennbar ist, gilt die Anforderung als erfüllt.
Beobachtung: Aus den von der Sensoreinheit bereitgestellten Daten wird eine Punktwolke erstellt, in der die Oberflächengeometrie des Objekts sichtbar ist.
Beurteilung: Da die Sensoreinheit im Rahmen des Gesamtprozesses ohnehin Daten aufnimmt, aus denen eine Punktwolke generiert wird, ist die Erfassung der Oberflächengeometrie gewährleistet. Die Anforderung ist damit **erfüllt**.

HW-SE 2 (F)

Testgegenstand: Die Sensoreinheit erfasst die Textur des Objekts.
Durchführung: Der Prozess wird wie vorgesehen ausgeführt. Neben den Tiefeninformationen werden während der Aufnahme Farbdaten erfasst. Die aus Farbbildern und Tiefenwerten resultierende Punktwolke enthält Farbwerte für jeden Punkt.
Akzeptanzkriterium: Ist nach Abschluss des Prozesses eine texturierte Punktwolke vorhanden, aus der die Farbverläufe und Muster der Objektoberfläche erkennbar sind, gilt die Anforderung als erfüllt.
Beobachtung: Nach der Verarbeitung liegt eine Punktwolke vor, in der die Farbwerte jedes Punkts die Textur des Objekts widerspiegeln.
Beurteilung: Da neben den Tiefendaten auch Farbinformationen aufgenommen und in die Punktwolke integriert wurden, ist die Erfassung der Objekttextur gewährleistet. Die Anforderung ist damit **erfüllt**.

HW-SE 3 (F)

Testgegenstand: Die Sensoreinheit ermöglicht die freihändige Bewegung um das gesamte Objekt.
Durchführung: Während des Aufnahmeprozesses wird die Sensoreinheit ohne feste Montage von Hand um das Objekt herumgeführt. Der Bediener positioniert die Sensoreinheit in verschiedenen Winkeln und Entfernungen, um möglichst alle Seiten des Objekts aufzunehmen.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn die Sensoreinheit ohne externe Hilfsmittel oder starren Befestigungen so um das Objekt bewegt werden kann, dass alle relevanten Bereiche prinzipiell erfassbar sind.
Beobachtung: Der Bediener konnte die Sensoreinheit frei um das Objekt herumführen,

ohne auf zusätzliche Halterungen zurückzugreifen. Dadurch wurden alle Seiten des Objekts aufgenommen. Allerdings erwiesen sich bestimmte Bereiche als schwierig vollständig zu erfassen. Dieser Effekt trat beispielsweise an Stellen auf, die durch Abschattungen oder ungünstige Winkellagen vom Sensor nur unvollständig erfasst werden konnten. Eine Anpassung, etwa durch eine veränderte Kamerahalterung mit einer anderen Winkeleinstellung oder eine erhöhte Platzierung des Objekts, könnte dieses Problem mindern.
Beurteilung: Die Sensoreinheit ermöglicht zwar eine grundsätzlich freihändige Bewegung um das Objekt, dennoch sind nicht alle relevanten Bereiche ohne weiteres zugänglich. Unter Berücksichtigung dieser Einschränkungen gilt die Anforderung als **teilweise erfüllt**.

HW-SE 4 (NF)

Testgegenstand: Die Sensoreinheit wiegt einschließlich aller Komponenten weniger als 400 Gramm.
Durchführung: Die vollständig montierte Sensoreinheit (Kameras, Halterungen und angeschlossene Kabel) wird gewogen, um das Gesamtgewicht zu bestimmen.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn das gemessene Gesamtgewicht der Sensoreinheit unter 400 Gramm liegt.
Beobachtung: Bei der Messung beträgt das Gesamtgewicht der Sensoreinheit einschließlich aller notwendigen Komponenten etwa 329 Gramm.
Beurteilung: Da die Masse der Sensoreinheit unter dem geforderten Grenzwert von 400 Gramm liegt, ist die Anforderung als **erfüllt** anzusehen.

HW-SE 5 (NF)

Testgegenstand: Das System ist in der Lage, Objekte mit matten Oberflächen aus Metall oder Kunststoff zu erfassen, die über Schaltelemente und Anzeigen verfügen.
Durchführung: Als Testobjekt dient ein Autopilotpanel mit einer matt lackierten Front, auf der sich diverse Schaltelemente und Anzeigen befinden. Mit der Sensoreinheit wird das Panel von der Vorderseite aufgenommen. Aus den erfassten Daten wird anschließend eine Punktwolke generiert. Abbildung 7.1 zeigt das verwendete Autopilotpanel.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn in der resultierenden Punktwolke sowohl die Oberfläche des Panels als auch die Schaltelemente und Anzeigen

Abbildung 7.1: Als Testobjekt genutztes Autopilotpanel

eindeutig erkennbar sind.
Beobachtung: Nach Abschluss der Aufnahme und Verarbeitung liegt eine Punktwolke vor, in der die Objektoberfläche klar zu erkennen ist. Die Schaltelemente und Anzeigen sind sichtbar und können unterschieden werden. Es treten keine Hinweise auf, dass die unterschiedlichen Materialeigenschaften Probleme bei der Erfassung verursachen. Sowohl die lackierte Front des Geräts als auch die unlackierte Metalloberseite werden erfolgreich erfasst. Lediglich die Bereiche unterhalb der Drehschalterkappen konnten aus der gewählten Perspektive aufgrund von Abschattungen nicht vollständig aufgenommen werden. Abbildung 7.2 zeigt die resultierende Punktwolke.
Beurteilung: Das System erfasst sowohl die matte Metall- oder Kunststoffoberfläche als auch komplexe Applikationen wie Schaltelemente und Anzeigen. Die Anforderung gilt somit als **erfüllt**.

HW-SE 6 (F, Optional)

Testgegenstand: Der Zustand des Scanvorgangs kann über Bedienelemente an der Sensoreinheit gesteuert werden.
In dieser Arbeit wurde diese Anforderung nicht umgesetzt, sodass der Scanvorgang weiterhin ausschließlich über den Laptop gesteuert wird. Während des Projektverlaufs wurden jedoch bereits verschiedene Mikrocontroller (siehe Abbildung 7.3) beschafft und hinsichtlich ihrer Eignung verglichen, um ein solches System prinzipiell zu ermöglichen. Die grundlegende Idee bestand darin, einen Mikrocontroller im Handstück der Sensoreinheit zu integrieren und diesen über einen der hinausgeführten Pins der Kamera mit Spannung zu versorgen. Über eine Funkverbindung sollte der Mikrocontroller anschließend Befehle zur Steuerung sowie Rückmeldungen zur Prozessüberwachung an den

Abbildung 7.2: Erzeugte Punktwolke des Autopilotpanels

Abbildung 7.3: Verglichene Mikrocontroller-Modelle des Herstellers *Seeed Studio*: XIAO-ESP32-S3 und XIAO-nRF52840

Laptop übermitteln. Dieser Ansatz wurde nicht weiterverfolgt, da der Schwerpunkt auf anderen Komponenten des Systems lag und der 3,3 V-Pin der Kamera nicht ausreichend dokumentiert war. Ohne hinreichende Informationen über die elektrische Auslegung dieses Pins wurde ein Anschluss daran als potenzielles Risiko für die Kamerahardware eingestuft.

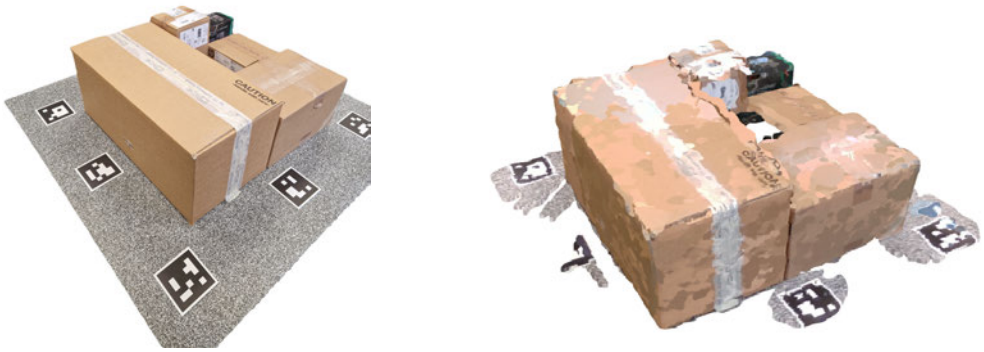
HW-SE 7 (F, Optional)

Testgegenstand: Die Sensoreinheit verfügt über eine integrierte Beleuchtungseinrichtung.
Diese Anforderung wurde im Rahmen dieser Arbeit nicht umgesetzt. Es traten ähnliche Probleme auf wie bei der zuvor beschriebenen geplanten Integration von Bedienelementen. Eine ausreichende Spannungsversorgung für leistungsstarke Leuchtmittel wäre ohne zusätzliche Kabel, welche die Ergonomie beeinträchtigen könnten, nicht realisierbar gewesen. Darüber hinaus würde der Einsatz einer Beleuchtungseinrichtung das Gesamtgewicht der Sensoreinheit weiter erhöhen und näher an die vorgegebene maximale Gewichtsgrenze bringen. Um die Aufnahmequalität dennoch zu verbessern, kann stattdessen auf externe Beleuchtungsmittel zurückgegriffen werden, ohne die Handhabung oder das Gewicht der Sensoreinheit nachteilig zu beeinflussen.

7.1.3 Anforderungen Aufnahmebereich

HW-AB 1 (NF)

Testgegenstand: Das System unterstützt einen Aufnahmebereich um Objekte mit den maximalen Abmessungen von 60 cm × 60 cm × 25 cm aufzunehmen.
Durchführung: Zur Überprüfung dieser Anforderung werden mehrere Objekte so im Aufnahmebereich platziert, dass sie gemeinsam einen Bereich von etwa 60 cm × 60 cm einschließen. Zudem wird sichergestellt, dass diese eine minimale Höhe von 25 cm abdecken. Die ArUco-Marker werden außerhalb dieses Bereichs ausgelegt. Anschließend wird die Sensoreinheit um den Aufnahmebereich geführt, um eine Punktwolke des Szenarios zu erzeugen. In Abbildung 7.4a ist der mit den Testobjekten belegte Aufnahmebereich dargestellt.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn alle Objekte, die zusammen der maximalen Objektgröße entsprechen, aus allen relevanten Perspektiven erfasst werden können und in der resultierenden Punktwolke abgebildet sind.
Beobachtung: Die erzeugte Punktwolke (siehe Abbildung 7.4b) zeigt alle im Aufnahmebereich platzierten Objekte. Das System ist in der Lage, auch noch größere Objekte aufzunehmen, sofern entsprechend angepasste Rauschmuster als Unterlage eingesetzt werden. Ist dies aufgrund der Objektgröße nicht möglich, kann das Objekt auch ohne Unterlage erfasst werden, wobei das nicht empfohlen wird. Grundsätzlich gilt, dass



(a) Aufnahmebereich mit ausgelegten Objekten

(b) Erzeugte Punktwolke

Abbildung 7.4: Überprüfen der Größe des Aufnahmebereichs

mindestens ein bekannter Marker im Bild identifiziert werden muss. Falls erforderlich, können zusätzliche Marker gedruckt und verwendet werden.
Beurteilung: Das System kann den für die vorgegebenen Maximalabmessungen erforderlichen Aufnahmebereich abdecken. Die Anforderung wird somit als **erfüllt** bewertet.

7.2 Softwareanforderungen

In diesem Abschnitt werden die Softwareanforderungen überprüft. Die Überprüfung erfolgt wieder separat für die einzelnen Teilabschnitte: generelle Anforderungen, Anforderungen an die Datenaufnahme, Anforderungen an die Datenverarbeitung und Anforderungen an die Bedienoberfläche.

7.2.1 Generelle Anforderungen

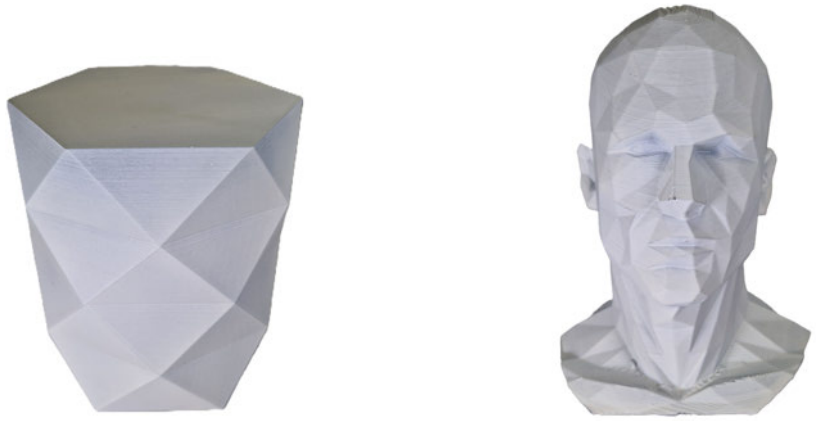
SW-GE 1 (NF)

Testgegenstand: Jedes Modul der Software muss über eine aussagekräftige Dokumentation verfügen.
Durchführung: In Python werden Module, Funktionen und Klassen mit Docstrings beschrieben, die eine Kurzbeschreibung des jeweiligen Objekts enthalten [42]. Zur Überprüfung wird für jedes Modul, jede Klasse und jede öffentliche Methode kontrolliert, ob

ein Docstring vorhanden ist.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn alle Klassen Methoden über einen Docstring verfügen.
Beobachtung: Alle Klassen und öffentlichen Methoden sind mit Docstrings versehen, die deren Funktionalität und Zweck beschreiben.
Beurteilung: Die Anforderung wird als **erfüllt** bewertet.

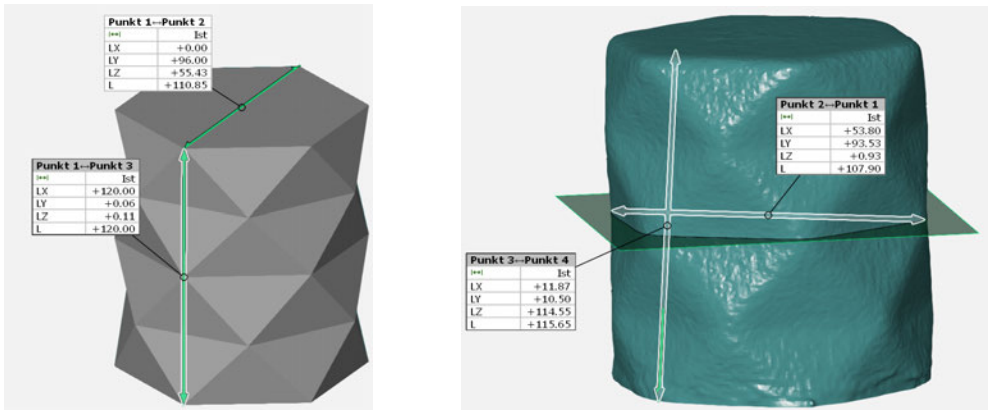
SW-GE 2 (NF)

Testgegenstand: Das System erzeugt digitale Abbilder von realen Objekten mit einer maximalen Abweichung von ±0,2 mm zum realen Objekt.
Durchführung: Zur Überprüfung der Genauigkeit werden zwei Aspekte betrachtet: Zum einen die Maßhaltigkeit hinsichtlich der globalen Dimensionen eines gescannten Objekts, zum anderen die Detailgenauigkeit bei der Erfassung feiner Strukturen.
Als Referenzobjekt für die Maßhaltigkeit dient ein Stifthalter [75], dessen Mantelfläche aus symmetrisch angeordneten Dreiecken besteht. Das Modell wird mit einem 3D-Drucker gefertigt, weiß matt lackiert (Abbildung 7.5a) und anschließend gescannt. Nach der Rekonstruktion werden dessen Abmessungen mithilfe der Software *GOM Inspect* bestimmt und mit den Originalmaßen verglichen.
Für die Beurteilung der Detailgenauigkeit wird ein Kopfmodell [43] verwendet, ebenfalls 3D-gedruckt, lackiert (Abbildung 7.5b) und gescannt. Hier wird das erzeugte Modell erneut ausgedruckt und anschließend visuell mit dem zuvor gedruckten Original verglichen.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn die gemessenen Abweichungen bei der Maßhaltigkeit maximal ±0,2 mm betragen. Zudem soll das gescannte und neu ausgedruckte Kopfmodell gegenüber dem zuvor gedruckten Originalmodell visuell nicht unterscheidbar sein, wobei angenommen wird, dass Abweichungen unter 0,2 mm im realen Vergleich nicht erkennbar sind.
Beobachtung: In Abbildung 7.6 sind die Messungen der Stifthaltermodelle dargestellt. Das Orignalmodell hat eine Breite von 110,85 mm und eine Höhe von 120 mm. Die Deckfläche des gescannten Objekts weist eine nicht unerhebliche Abrundung auf. Daher wird das Objekt entlang dessen Höhe an einer Stelle geschnitten, die der gleichen Fläche wie die Deckfläche entspricht. An der korrespondierenden Stelle weist das gescannte Modell eine Breite von 107,9 mm auf. Die Höhe wird anhand eines Punktes auf der Deckfläche und einem der untersten Punkte der Mantelfläche ermittelt. Von diesem Abstand wird



(a) Testobjekt Stifthalter (b) Testobjekt Kopf

Abbildung 7.5: Zur Einschätzung der Genauigkeit verwendete Testobjekte



(a) Messungen am Originalmodell (b) Messungen am gescannten Modell

Abbildung 7.6: Vergleich der Messungen von Original- und gescanntem Stifthalter

lediglich die Z-Komponente betrachtet. Demnach hat das gescannte Modell eine Höhe von 115,65 mm. Beim Kopfmodell kann die grundlegende Geometrie (beispielsweise die Kopfform) im gescannten Modell noch erkannt werden, feine Details sind jedoch verloren gegangen. Der optische Vergleich zwischen Original und gescanntem Modell zeigt, dass sich die Modelle visuell eindeutig unterscheiden lassen. Abbildung 7.7 zeigt beide Modelle im direkten Vergleich.

Beurteilung: Die gemessenen Abweichungen der globalen Abmessungen des Stifthalters liegen deutlich über dem vorgegebenen Toleranzbereich von ±0,2 mm. Zudem konnte beim Kopfmodell die erforderliche Detailgenauigkeit nicht erreicht werden, sodass feinere Merkmale nicht in ausreichender Qualität erfasst wurden. Um die Anforderun-

Abbildung 7.7: Vergleich beider Kopfmodelle: Links das Originalmodell und Rechts das gescannte und gedruckte Modell

gen erfüllen zu können, müssten Kameras mit einer höheren Tiefenauflösung eingesetzt werden als die hier verwendeten, um die feinen Details präzise aufnehmen zu können. Die globalen Abmessungen bieten jedoch bereits eine zufriedenstellende Genauigkeit für praktische Anwendungen.
Insgesamt erfüllt das hier entwickelte System die geforderten Genauigkeitskriterien nicht. Die Anforderung wird daher als **nicht erfüllt** bewertet.

SW-GE 3 (F)

Testgegenstand: Die Software ist auf einem Computer mit dem Betriebssystem Windows 10 lauffähig.
Durchführung: Die Software wird auf einem Laptop mit Windows 10 gestartet. Anschließend wird ein kompletter Scanprozess durchgeführt.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn die Software ohne Fehlermeldungen ausgeführt und der gesamte Scanprozess erfolgreich abgeschlossen werden kann.

Beobachtung: Während des Starts und der Nutzung der Software unter Windows 10 traten keinerlei Fehlermeldungen auf. Der gesamte Scanprozess verlief reibungslos.
Beurteilung: Die Anforderung wird als **erfüllt** bewertet.

SW-GE 4 (NF)

Testgegenstand: Die Datenverarbeitung vom Abschluss der Aufnahme bis zur Ausgabe des fertigen Modells soll auf einem Computer mit einem Windows-Leistungsindex von mindestens sieben innerhalb von maximal $2+\frac{n}{2}$ Minuten abgeschlossen sein, wobei n der Anzahl der aufgenommenen Perspektiven entspricht.
Durchführung: Auf einem geeigneten Windows-System mit einem Leistungsindex von mindestens 7 wird ein Testobjekt aus 48 Perspektiven aufgenommen. Diese Anzahl hat sich im Verlauf der Untersuchungen mit den vorhandenen Testobjekten als geeigneter Richtwert erwiesen, um qualitativ gute Ergebnisse zu erzielen. Anschließend wird die benötigte Zeit für die Vorverarbeitung, Registrierung, Rekonstruktion und den Export des resultierenden Modells gemessen.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn die Verarbeitungsdauer 26 Minuten nicht überschreitet ($n = 48$, $2 + \frac{48}{2} = 26$).
Beobachtung: Der gesamte Verarbeitungsprozess wurde nach etwa 21 Minuten erfolgreich abgeschlossen.
Beurteilung: Die Anforderung wird als **erfüllt** bewertet.

7.2.2 Anforderungen an die Datenaufnahme

SW-DA 1 (F)

Testgegenstand: Die Änderung der Kameraposition zwischen den aufgenommenen Perspektiven kann nach der Datenaufnahme ermittelt werden.
Durchführung: Es werden zwei Perspektiven aufgenommen, bei denen die Kamera von der Mitte einer Kante des Aufnahmebereichs zur Mitte einer angrenzenden Kante bewegt wird. Anschließend wird anhand der ArUco-Marker die Transformationsmatrix zwischen diesen beiden Perspektiven bestimmt und ausgewertet. Dabei wird für Roll (Rotation um die Z-Achse), Pitch (Rotation um die X-Achse) und Yaw (Rotation um die Y-Achse) jeweils der Rotationswinkel berechnet, sowie die Translation in Metern

entlang der drei Achsen ermittelt. Erwartet wird in diesem Test eine Rotation von etwa 90 Grad um die Y-Achse sowie eine Translation von ca. 50 cm entlang der X- und Z-Achse.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn die berechneten Rotations- und Translationswerte qualitativ den erwarteten Werten entsprechen.
Beobachtung: Die ausgegebene Rotation betrug etwa 11,38° um die X-Achse, 90,6° um die Y-Achse und 8,89° um die Z-Achse. Für die Translation wurden etwa eine Änderung um 57 cm entlang der X-Achse, 4,5 cm entlang der Y-Achse sowie 58 cm entlang der Z-Achse ermittelt.
Beurteilung: Die ermittelten Transformationswerte stimmen qualitativ mit den erwarteten Änderungen überein. Die vorhandenen Ungenauigkeit stammen aller Wahrscheinlichkeit nach von der manuellen Platzierung der Sensoreinheit. Somit wird die Anforderung als **erfüllt** bewertet.

SW-DA 2 (F, Optional)

Testgegenstand: Die Beleuchtungseinrichtung kann während der Datenaufnahme vom steuernden Computer in ihrer Intensität angepasst werden, um das Objekt aus allen Richtungen gleichmäßig auszuleuchten.
Da im Rahmen dieser Arbeit keine Beleuchtungseinrichtung in die Sensoreinheit integriert wurde, bestand keine Möglichkeit, deren Intensität softwareseitig zu regulieren. Diese Anforderung wurde somit nicht bearbeitet und folglich auch nicht erfüllt.

SW-DA 3 (F, Optional)

Testgegenstand: Während der Datenaufnahme erhält der Benutzer Feedback zum aktuellen Abstand der Sensoreinheit zum Objekt.
Diese Anforderung wurde im Rahmen dieser Arbeit nicht umgesetzt. Mit der entwickelten Software wäre eine solche Funktionalität allerdings durchaus sinnvoll. Bei einem zu geringen Abstand zum Sensor werden unzulässige Daten ausgegeben. Eine Überprüfung der Anteile der Daten die unzulässig sind, könnte damit als Einschätzung dienen, ob der Abstand zu gering ist. Andersherum, kann der mittlere Abstand zum Objekt aus den zulässigen Daten ermittelt werden und Rückschluss drüber geben, wenn die Sensoreinheit näher an das Objekt gebracht werden könnte. Die dafür notwendigen Algorithmen

zur laufenden Auswertung der Daten wurden jedoch zugunsten anderer Schwerpunkte nicht implementiert.

SW-DA 4 (F, Optional)

Testgegenstand: Während der Datenaufnahme erhält der Benutzer Feedback zum aktuellen Fortschritt des Scanvorgangs.
Zwar erhält der Benutzer während der Datenaufnahme grundlegende Informationen darüber, wie viele Datensätze bereits aufgezeichnet wurden und wie viele noch erforderlich sind. Die ursprünglich geplante Funktionalität, eine visuelle Darstellung der aufgenommenen Bereiche bereitzustellen, um gezielt Regionen mit geringer Punktdichte zu identifizieren, wurde jedoch nicht umgesetzt. Stattdessen bleibt es bei einer einfachen numerischen Fortschrittsanzeige.

7.2.3 Anforderungen an die Datenverarbeitung

SW-DV 1 (F) und SW-DV 2 (F)

Testgegenstand: Die Software speichert Objekte zur Reproduktion mittels 3D-Drucker im *.stl*-Format.
Die Software speichert Objekte zur digitalen Weiterverarbeitung im `.obj`-Format.
Durchführung: Der Aufnahme- und Verarbeitungsprozess wird bis zum fertigen Modell vollständig durchlaufen. Anschließend wird das resultierende Objekt einmal als *.stl*- und einmal als *.obj*-Datei exportiert. Beide Dateien werden mit einem externen Viewer geöffnet, um zu überprüfen, ob das Modell korrekt gespeichert wurde und dargestellt werden kann.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn beide exportierten Dateien ohne Fehlermeldungen geladen werden können und das erwartete 3D-Modell enthalten.
Beobachtung: Sowohl die *.stl*- als auch die *.obj*-Datei konnten erfolgreich geöffnet werden. In beiden Fällen war das entsprechende Objekt vollständig und korrekt dargestellt.
Beurteilung: Die Anforderung wird als **erfüllt** bewertet.

7.2.4 Anforderungen an die Bedienoberfläche

SW-BO 1 (F)

Testgegenstand: Der Prozess der 3D-Modellerstellung wird durch eine grafische Benutzeroberfläche gesteuert.
Durchführung: Während der Aufnahme, Verarbeitung und Rekonstruktion des 3D-Modells wird überprüft, ob alle wesentlichen Schritte (beispielsweise Starten der Aufnahme, Vorverarbeitung, Feinregistrierung, Rekonstruktion und Export) über ein GUI ausgelöst und gesteuert werden können.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn der Benutzer ohne direkte Eingriffe in den Quellcode oder die Kommandozeile alle relevanten Prozesse der 3D-Modellerstellung über die GUI starten und steuern kann.
Beobachtung: Alle wesentlichen Verfahrensschritte sind in das GUI integriert. Der Benutzer kann über Schaltflächen und Dialoge die Datenaufnahme starten, Filter- und Registrierungsprozesse ausführen, die Rekonstruktion anstoßen sowie das fertige Modell exportieren. Ein direkter Zugriff auf die Kommandozeile ist nicht erforderlich.
Beurteilung: Da die gesamte Prozesskette über die GUI gesteuert werden kann, wird die Anforderung als **erfüllt** bewertet.

SW-BO 2 (F)

Testgegenstand: Um ein Bewusstsein für den aktuellen Programmstand zu schaffen, werden in jedem Schritt des Programms die aktuellen Daten in einer 2D- oder 3D-Anzeige dargestellt.
Durchführung: Während des Programmablaufs werden alle Verarbeitungsschritte (z.B. Aufnahme, Vorverarbeitung, Registrierung, Rekonstruktion) überprüft. Dabei wird kontrolliert, ob zu jedem Verarbeitungsschritt im GUI eine entsprechende Visualisierung der aktuellen Daten erfolgt.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn der Benutzer zu jedem Schritt eine visuelle Darstellung erhält, aus der der Prozessfortschritt sowie die Art der gerade bearbeiteten Daten hervorgehen.
Beobachtung: Während der Ausführung werden bei der Aufnahme Farbbilder angezeigt, in der Vorverarbeitungs- und Registrierungsphase werden Punktwolken in einer 3D-Ansicht präsentiert, und während der Rekonstruktion ist das entstehende Modell

sichtbar.
Beurteilung: Da in jedem Programmschritt eine geeignete visuelle Darstellung der aktuellen Daten erfolgt, wird die Anforderung als **erfüllt** bewertet.

SW-BO 3 (NF)

Testgegenstand: Die angezeigten dreidimensionalen Daten können rotiert und skaliert werden.
Durchführung: Während der Programmausführung werden die 3D-Daten (z.B. Punktwolken oder rekonstruierte Modelle) im GUI dargestellt. Es wird überprüft, ob der Benutzer mithilfe geeigneter Interaktionen (z.B. Mausbewegungen, Mausrad) die Darstellung frei drehen und deren Maßstab verändern kann.
Akzeptanzkriterium: Die Anforderung gilt als erfüllt, wenn der Benutzer die 3D-Darstellung in beliebige Richtungen rotieren und deren Größe anpassen kann.
Beobachtung: Die im GUI integrierte Anzeige ermöglicht es, die 3D-Daten durch entsprechende Mausbewegungen zu drehen und mit dem Mausrad zu skalieren.
Beurteilung: Da der Benutzer die angezeigten dreidimensionalen Daten sowohl rotieren als auch skalieren kann, wird die Anforderung als **erfüllt** bewertet.

8 Fazit und Ausblick

Die vorliegende Arbeit verfolgte das Ziel, einen 3D-Scanner zu entwickeln, der aus Aufnahmen einer mobilen Sensoreinheit dreidimensionale Modelle von Objekten erzeugt. Hierfür sollten sowohl die Hardware- als auch die Softwarekomponenten so konzipiert und aufeinander abgestimmt werden, dass ein weitgehend automatisierter Prozess entsteht, der ohne tiefgreifendes Fachwissen nutzbar ist.
Im Rahmen des Projekts konnte ein funktionsfähiger Prototyp realisiert werden, der sämtliche Prozessschritte – von der Aufnahme und Vorverarbeitung über die Registrierung und Oberflächenrekonstruktion bis hin zum finalen Export des 3D-Modells – durchführt. Die erzielten Ergebnisse zeigten jedoch, dass das System die selbst gesetzten Anforderungen an Genauigkeit und Robustheit noch nicht vollständig erfüllt. Insbesondere die erreichte Modellgenauigkeit liegt außerhalb der angestrebten Toleranzen, und der Aufnahmeprozess erweist sich als stark von äußeren Faktoren abhängig. Dabei spielen Eigenschaften des zu erfassenden Objekts, wie beispielsweise Material, Farbe oder Oberflächenbeschaffenheit, eine zentrale Rolle für die Güte der Resultate. Demnach wurde die anvisierte Entwicklung eines präzisen und zuverlässigen Systems nur teilweise realisiert.
Die Entscheidung, zu Beginn der Arbeit umfassende Anforderungen und klare Zielsetzungen festzulegen, hat sich als sinnvoll erwiesen. Diese Vorgehensweise ermöglichte eine zielgerichtete Planung und eine klare Fokussierung auf die wesentlichen Systemkomponenten. Dennoch zeigte sich im Projektverlauf, dass insbesondere in den Bereichen Registrierung und Oberflächenrekonstruktion unerwartete Herausforderungen aufgetreten sind, die viele Tests und Anpassungen erfordern. Für künftige Projekte empfiehlt es sich, frühzeitig verschiedene Algorithmen und Methoden zu erproben und dabei stets das übergeordnete Ziel im Blick zu behalten, um nicht in zeitaufwändige Detailoptimierungen abzudriften.
Im Hinblick auf zukünftige Arbeiten bieten sich verschiedene Ansätze zur Verbesserung an. Zum einen ließe sich die Genauigkeit des Systems durch erweiterte Filter- und Glättungsmethoden, insbesondere an den Rändern der Punktwolken, steigern. Zum anderen

könnten individuell angepasste Rekonstruktionsalgorithmen entwickelt werden. Auch der Einsatz verbesserter oder zusätzlicher Sensorik, etwa für eine optimierte Beleuchtung oder eine automatisierte Abstandsmessung, könnte den Bedienkomfort und die Qualität der rekonstruierten Modelle erhöhen.
Hinsichtlich der beiden Anwendungsfälle – Objektreproduktion und Automatisierung von Testabläufen – eignet sich das System vor allem für die Reproduktion einfacher Geometrien, sofern die Oberflächenbeschaffenheit des Objekts geeignet ist. Bei komplexen Strukturen und feinen Details, wie sie bei Geräten für Testabläufe mit Schaltelementen und Anzeigen auftreten, liefern die bisherigen Ansätze noch keine zufriedenstellenden Ergebnisse. Durch eine weiterführende Nachbearbeitung der Daten wäre es jedoch denkbar, diese auch für derartige Anwendungen nutzbar zu machen. Entsprechende Untersuchungen und Verbesserungen lassen sich in zukünftigen Arbeiten umsetzen.
Das hier vorgestellte System legt somit eine solide Grundlage, auf der zukünftige Projekte aufbauen können, um die Genauigkeit, Robustheit und Anwendungsvielfalt des Scanners weiter zu steigern.

Literaturverzeichnis

[1] 3D, Creality: *CR-Scan Ferret*. https://www.creality.com/de/products/cr-scan-ferret. – Zugriffsdatum: 11.10.2024

[2] 3D, Creality: *CR-Scan Ferret 3D Scanner - User Manual*. https://wiki.creality.com/en/3d-scanner/cr-scan-ferret/manual. – Zugriffsdatum: 11.10.2024

[3] 3D, Creality: *CR-Scan Ferret SE*. https://store.creality.com/de/products/cr-scan-ferret-se. – Zugriffsdatum: 11.10.2024

[4] 3D, Creality: *CR-Scan Raptor*. https://store.creality.com/de/products/cr-scan-raptor. – Zugriffsdatum: 07.11.2024

[5] 3D, SHINING: *EinScan Color Pack*. https://www.einscan.com/handheld-3d-scanner/color-pack/. – Zugriffsdatum: 07.11.2024

[6] 3D, SHINING: *EinScan Pro HD HIGH DEFINITION, MULTI-FUNCTIONAL HANDHELD 3D SCANNER*. https://www.einscan.com/wp-content/uploads/2020/06/EinScan-PRO-HDenV0.101.pdf. – Zugriffsdatum: 11.10.2024

[7] Arbeitsstätten ASTA-Geschäftsführung, Ausschuss für: *Technische Regeln für Arbeitsstätten: ASR A1.5 Fußböden*. https://www.baua.de/DE/Angebote/Regelwerk/ASR/pdf/ASR-A1-5.pdf?__blob=publicationFile&v=6. – Zugriffsdatum: 06.12.2024

[8] Artec 3D: *Eva helps produce a 3D-printed prosthesis of the gastrocnemius muscle*. https://www.artec3d.com/news/eva-helps-produce-3d-printed-prosthesis-gastrocnemius-muscle. – Zugriffsdatum: 11.10.2024

[9] Benjemaa, R. ; Schmitt, F.: Fast global registration of 3D sampled surfaces using a multi-z-buffer technique. In: *Proceedings. International Conference on Recent Advances in 3-D Digital Imaging and Modeling (Cat. No.97TB100134)*, 1997, S. 113–120

[10] Bernardini, F. ; Mittleman, J. ; Rushmeier, H. ; Silva, C. ; Taubin, G.: The ball-pivoting algorithm for surface reconstruction. In: *IEEE Transactions on Visualization and Computer Graphics* 5 (1999), Nr. 4, S. 349–359

[11] BerryBase: *Stereo Kamera Modul IMX219-83, 8 Megapixel*. https://www.berrybase.de/stereo-kamera-modul-imx219-83-8-megapixel. – Zugriffsdatum: 13.11.2024

[12] Besl, P.J. ; McKay, Neil D.: A method for registration of 3-D shapes. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14 (1992), Nr. 2, S. 239–256

[13] Beyerer, Jürgen ; León, Fernando P. ; Frese, Christian: *Automatische Sichtprüfung: Grundlagen, Methoden und Praxis der BIldgewinnung und Bildauswertung*. Springer Berlin Heidelberg, 2012. – URL https://www.google.de/books/edition/Automatische_Sichtpr%C3%BCfung/-TIiBAAAQBAJ?hl=de&gbpv=0. – ISBN 978-3-642-23965-6

[14] Blais, G. ; Levine, M.D.: Registering multiview range data to create 3D computer objects. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17 (1995), Nr. 8, S. 820–824

[15] Calonder, Michael ; Lepetit, Vincent ; Strecha, Christoph ; Fua, Pascal V.: BRIEF: Binary Robust Independent Elementary Features. In: *European Conference on Computer Vision*, URL https://api.semanticscholar.org/CorpusID:1815530, 2010

[16] Carfagni, Monica ; Furferi, Rocco ; Governi, Lapo ; Santarelli, Chiara ; Servi, Michaela ; Uccheddu, Francesca ; Volpe, Yary: Metrological and Critical Characterization of the Intel D415 Stereo Depth Camera. In: *Sensors, Volume 19, Issue 3* (2019), Januar

[17] Chen, Yang ; Medioni, Gérard: Object modelling by registration of multiple range images. In: *Image and Vision Computing* 10 (1992), Nr. 3, S. 145–155. – URL

https://www.sciencedirect.com/science/article/pii/026288569290066C. – ISSN 0262-8856

[18] Choi, Sungjoon ; Zhou, Qian-Yi ; Koltun, Vladlen: Robust reconstruction of indoor scenes. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, S. 5556–5565

[19] Computing, Riverbank: *PyQT*. https://riverbankcomputing.com/software/pyqt/intro. – Zugriffsdatum: 14.11.2024

[20] Corporation, Intel: *D400 Series Product Family*. https://www.intel.com/content/dam/support/us/en/documents/emerging-technologies/intel-realsense-technology/Intel-RealSense-D400-Series-Datasheet.pdf. – Zugriffsdatum: 27.08.2024

[21] Das, Gopal: *k-d Tree and Nearest Neighbor Search*. https://gopalcdas.com/2017/05/24/construction-of-k-d-tree-and-using-it-for-nearest-neighbour-search/. – Zugriffsdatum: 01.10.2024

[22] Dorai, C. ; Wang, G. ; Jain, A.K. ; Mercer, C.: Registration and integration of multiple object views for 3D model construction. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20 (1998), Nr. 1, S. 83–89

[23] Edelsbrunner, H. ; Kirkpatrick, D. ; Seidel, R.: On the shape of a set of points in the plane. In: *IEEE Transactions on Information Theory* 29 (1983), Nr. 4, S. 551–559

[24] Eggert, D.W. ; Lorusso, Alice ; Fisher, Robert: Estimating 3-D Rigid Body Transformations: A Comparison of Four Major Algorithms. In: *Machine Vision and Applications* 9 (1997), 03, S. 272–290

[25] electronics, SKALARKI: *Skalarki FCU*. https://skalarki-electronics.com/. – Zugriffsdatum: 15.10.2024

[26] Emmerich, Robert: *3D-Scans für die Automobil-Industrie*. https://www.uni-wuerzburg.de/aktuelles/pressemitteilungen/single/news/3d-scans-fuer-die-automobil-industrie-1/. – Zugriffsdatum: 11.10.2024

[27] Engelmann, Francis: *The FabScan Project*. https://hci.rwth-aachen.de/fabscan. – Zugriffsdatum: 27.05.2024

[28] Engelmann, Francis: *FabScan - Affordable 3D Laser Scanning of Physical Objects*, RWTH Aachen, Masterarbeit, 2011. – URL https://hci.rwth-aachen.de/publications/engelmann2011a.pdf

[29] FILAMENT, DAS: *PLA Filament - 1,75 mm - Grau - 800 g*. https://www.dasfilament.de/filament-spulen/pla-1-75-mm/8/pla-filament-1-75-mm-grau?c=11. – Zugriffsdatum: 01.07.2024

[30] fionec GmbH: *Tactile And Optical Measurement Technology Advantages, differences, and application fields*. https://www.fionec.com/english/applications/optical-or-tactile/. – Zugriffsdatum: 25.09.2024

[31] Fischer, Kaspar: An Introduction to Alpha Shapes / ETH Zürich. URL https://graphics.stanford.edu/courses/cs268-11-spring/handouts/AlphaShapes/as_fisher.pdf, 2005. – Forschungsbericht. Technical Report

[32] Fischler, Martin A. ; Bolles, Robert C.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. In: Fischler, Martin A. (Hrsg.) ; Firschein, Oscar (Hrsg.): *Readings in Computer Vision*. San Francisco (CA) : Morgan Kaufmann, 1987, S. 726–740. – URL https://www.sciencedirect.com/science/article/pii/B9780080515816500702. – ISBN 978-0-08-051581-6

[33] Foundation, Free S.: *How to Use GNU Licenses for Your Own Software*. https://www.gnu.org/licenses/gpl-howto.html.en. – Zugriffsdatum: 14.11.2024

[34] Fratz, Markus ; al. et: *Leitfaden zur optischen 3D-Messtechnik*. Fraunhofer-Verlag, 2014. – URL https://www.google.de/books/edition/_/l3DzoQEACAAJ?hl=de&sa=X&ved=2ahUKEwjBhaysm_mFAxUJhv0HHU7ADSQQre8FegQIDRAD. – ISBN 978-3-8396-0761-9

[35] Gadelha, Matheus ; Wang, Rui ; Maji, Subhransu: Multiresolution Tree Networks for 3D Point Cloud Processing. In: *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part VII*. Berlin, Heidelberg : Springer-Verlag, 2018, S. 105–122. – URL https://doi.org/10.1007/978-3-030-01234-2_7. – ISBN 978-3-030-01233-5

[36] Garrido-Jurado, S. ; Muñoz-Salinas, R. ; Madrid-Cuevas, F.J. ; Marín-Jiménez, M.J.: Automatic generation and detection of highly reliable fiducial markers under occlusion. In: *Pattern Recognition* 47 (2014), Nr. 6, S. 2280–2292. – URL https://www.sciencedirect.com/science/article/pii/S0031320314000235. – ISSN 0031-3203

[37] Giancola, Silvio ; Valenti, Matteo ; Sala, Remo: *A Survey on 3D Cameras: Metrological Comparison of Time-of-Flight, Structured-Light and Active Stereoscopy Technologies*. Springer International Publishing, 2018. – URL https://www.google.de/books/edition/A_Survey_on_3D_Cameras_Metrological_Comp/FutgDwAAQBAJ?hl=de&gbpv=0. – ISBN 978-3-319-91761-0

[38] GmbH, Carl Zeiss Industrielle M.: *Tactile And Optical Measurement Technology Advantages, differences, and application fields*. https://www.zeiss.com/metrology/en/explore/topics/tactile-and-optical-measurement-technology.html. – Zugriffsdatum: 25.09.2024

[39] GmbH, Scankraft: *Shining 3D EinScan Pro HD*. https://www.scankraft.com/produkt/shining-3d-einscan-pro-hd/. – Zugriffsdatum: 11.10.2024

[40] Godin, Guy ; Rioux, Marc ; Baribeau, Rejean: Three-dimensional registration using range and intensity information. In: El-Hakim, Sabry F. (Hrsg.): *Videometrics III* Bd. 2350, Oktober 1994, S. 279–290

[41] Gollor, Friederike C.: *Entwicklung eines Systems für Oberflächenscans und die Erstellung von 3D-Modellen von Personen*, Hochschule für Angewandte Wissenschaften Hamburg, Masterarbeit, 2022. – URL http://hdl.handle.net/20.500.12738/14841

[42] Goodger, David ; Rossum, Guido van: *PEP 257 – Docstring Conventions*. https://peps.python.org/pep-0257/. – Zugriffsdatum: 12.12.2024

[43] GreenCopper: *Male Head Lowpoly - no support - real size*. https://www.thingiverse.com/thing:5670717. – Zugriffsdatum: 06.12.2024

[44] Grunnet-Jepsen, Anders ; N. Sweetser, John ; Khuong, Tri ; Tong, Dave ; Woodfill, John: *Subpixel Linearity Improvement for Intel® RealSense™ Depth Camera D400 Series*. https://dev.intelrealsense.com/docs/white-paper-subpixel-linearity-improvement-for-intel-realsense-depth-cameras. – Zugriffsdatum: 23.08.2024

[45] Grunnet-Jepsen, Anders ; N. Sweetser, John ; Woodfill, John: *Tuning depth cameras for best performance*. https://dev.intelrealsense.com/docs/tuning-depth-cameras-for-best-performance. – Zugriffsdatum: 02.08.2024

[46] Gumhold, Stefan ; Wang, Xinlong ; MacLeod, Rob: Feature Extraction from Point Clouds. In: *Proceedings of 10th international meshing roundtable* 2001 (2001), 11

[47] Haider, Azmi ; Hel-Or, Hagit: What Can We Learn from Depth Camera Sensor Noise? In: *Sensors* 22 (2022), 07, S. 5448

[48] Hensel, Marc: *Informationen zu Abschlussarbeiten: Formalit¨aten, Aufbau, Hinweise*. 2023

[49] Hubeli, Andreas ; Gross, Markus: Multiresolution Feature Extraction from Unstructured Meshes. In: *Proc. IEEE Visualization 2001* 1 (2001), 01

[50] Kazhdan, Michael ; Bolitho, Matthew ; Hoppe, Hugues: Poisson surface reconstruction. In: *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*. Goslar, DEU : Eurographics Association, 2006 (SGP '06), S. 61–70. – ISBN 3905673363

[51] Kazhdan, Michael M. ; Bolitho, Matthew ; Hoppe, Hugues: Poisson surface reconstruction. In: *Eurographics Symposium on Geometry Processing*, URL https://api.semanticscholar.org/CorpusID:14224, 2006

[52] Koneva, Varvara: *Artec Eva helps to create cutting-edge face prostheses for children with severe burns*. https://www.artec3d.com/cases/romans-ferrari-3d-printed-masks. – Zugriffsdatum: 11.10.2024

[53] Kümmerle, Rainer ; Grisetti, Giorgio ; Strasdat, Hauke ; Konolige, Kurt ; Burgard, Wolfram: G2o: A general framework for graph optimization. In: *2011 IEEE International Conference on Robotics and Automation*, 2011, S. 3607–3613

[54] Lawson, Charles L.: *Software for C1 Surface Interpolation*. 1977. – URL https://api.semanticscholar.org/CorpusID:118951457

[55] Levoy, M.: The digital Michelangelo project. In: *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*, 1999, S. 2–11

[56] Lindqvist, Bo ; Skogsberg, Lars: *Ergonomie bei Handwerkzeugen*. Atlas Copco, 2014. – URL https://www.google.de/books/edition/Ergonomie_bei_Handwerkzeugen/pgQOjwEACAAJ?hl=de. – ISBN 978-91-633-2580-9

[57] Lorensen, William E. ; Cline, Harvey E.: Marching cubes: A high resolution 3D surface construction algorithm. In: *SIGGRAPH Comput. Graph.* 21 (1987), August, Nr. 4, S. 163–169. – URL https://doi.org/10.1145/37402.37422. – ISSN 0097-8930

[58] Lowe, David G.: Distinctive Image Features from Scale-Invariant Keypoints. In: *International Journal of Computer Vision* 60 (2004), S. 91–110. – URL https://api.semanticscholar.org/CorpusID:174065

[59] Lukas, Mario: *FabScan Pi - an open-hardware stand-alone web-enabled 3D scanner*, RWTH Aachen, Bachelorarbeit, 2015. – URL https://hci.rwth-aachen.de/publications/lukas2015a.pdf

[60] Maiti, Abhik ; Chakravarty, Debashish: Performance analysis of different surface reconstruction algorithms for 3D reconstruction of outdoor objects from their digital images. In: *SpringerPlus* 5 (2016), 06, S. 932

[61] Masuda, T. ; Sakaue, K. ; Yokoya, N.: Registration and integration of multiple range images for 3-D model construction. In: *Proceedings of 13th International Conference on Pattern Recognition* Bd. 1, 1996, S. 879–883 vol.1

[62] McMillion, Matthew: *Künstliche Ohren für Mikrotie-Patienten dank Artec Space Spider*. https://www.artec3d.com/de/cases/lewin-ear. – Zugriffsdatum: 11.10.2024

[63] Media, DIN: *DIN EN 60825-1:2022-07 - VDE 0837-1:2022-07*. https://www.dinmedia.de/de/norm/din-en-60825-1/344085844. – Zugriffsdatum: 16.10.2024

[64] Neugebauer, P.J.: Geometrical cloning of 3D objects via simultaneous registration of multiple range images. In: *Proceedings of 1997 International Conference on Shape Modeling and Applications*, 1997, S. 130–139

[65] Open3D: *GlobalOptimizationConvergenceCriteria.h*. https://github.com/isl-org/Open3D/blob/fcc396e494dd3f1204e53571f37400cac50976ef/cpp/open3d/pipelines/registration/GlobalOptimizationConvergenceCriteria.h. – Zugriffsdatum: 29.10.2024

[66] OPEN3D: *GlobalOptimization.h.* https://github.com/isl-org/Open3D/blob/fcc396e494dd3f1204e53571f37400cac50976ef/cpp/open3d/pipelines/registration/GlobalOptimization.h. – Zugriffsdatum: 29.10.2024

[67] OPEN3D: *Registration.h.* https://github.com/isl-org/Open3D/blob/main/cpp/open3d/t/pipelines/registration/Registration.h. – Zugriffsdatum: 22.10.2024

[68] OPEN3D: *Voxelization.* https://www.open3d.org/docs/latest/tutorial/Advanced/voxelization.html. – Zugriffsdatum: 26.09.2024

[69] OPENCV: *Detection of ChArUco Boards.* https://docs.opencv.org/3.4/df/d4a/tutorial_charuco_detection.html. – Zugriffsdatum: 05.09.2024

[70] OPENSCAN: *OPENSCAN.* https://www.openscan.eu/. – Zugriffsdatum: 27.05.2024

[71] OTEPKA, Johannes ; GHUFFAR, Sajid ; WALDHAUSER, Christoph ; HOCHREITER, Ronald ; PFEIFER, Norbert: Georeferenced Point Clouds: A Survey of Features and Point Cloud Management. In: *ISPRS Int. J. Geo-Inf, Volume 2, Issue 4* (2013), Oktober

[72] PARK, Jaesik ; ZHOU, Qian-Yi ; KOLTUN, Vladlen: Colored Point Cloud Registration Revisited. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, S. 143–152

[73] PULLI, K.: Multiview registration for large data sets. In: *Second International Conference on 3-D Digital Imaging and Modeling (Cat. No.PR00062)*, 1999, S. 160–168

[74] PULLI, Kari ; SHAPIRO, Linda G.: Surface Reconstruction and Display from Range and Color Data. In: *Graphical Models* 62 (2000), Nr. 3, S. 165–201. – URL https://www.sciencedirect.com/science/article/pii/S1524070399905192. – ISSN 1524-0703

[75] RICHARDGM: *Pencil Pen Holder Low Poly.* https://www.thingiverse.com/thing:3563343. – Zugriffsdatum: 06.12.2024

[76] ROSTEN, Edward ; DRUMMOND, Tom: Machine Learning for High-Speed Corner Detection. In: *European Conference on Computer Vision*, URL https://api.semanticscholar.org/CorpusID:1388140, 2006

[77] Rusinkiewicz, S. ; Levoy, M.: Efficient variants of the ICP algorithm. In: *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, 2001, S. 145–152

[78] Rusu, Radu ; Blodow, Nico ; Marton, Zoltan ; Beetz, Michael: Aligning Point Cloud Views using Persistent Feature Histograms. In: *IEEE/RSJ Int Conf Intel Robots Syst 2008,* (2008), 09, S. 3384–3391

[79] Rusu, Radu B. ; Blodow, Nico ; Beetz, Michael: Fast Point Feature Histograms (FPFH) for 3D registration. In: *2009 IEEE International Conference on Robotics and Automation*, 2009, S. 3212–3217

[80] Salti, Samuele ; Tombari, Federico ; Di Stefano, Luigi: SHOT: Unique signatures of histograms for surface and texture description. In: *Computer Vision and Image Understanding* 125 (2014), S. 251–264. – URL https://www.sciencedirect.com/science/article/pii/S1077314214000988. – ISSN 1077-3142

[81] SCANTECH: *3D-Scannen in der Automobilindustrie*. https://www.3d-scantech.com/de/applications/automobil/. – Zugriffsdatum: 11.10.2024

[82] Simon, David: *Fast and Accurate Shape-Based Registration*. Pittsburgh, PA, Carnegie Mellon University, Dissertation, December 1996

[83] Solutions, Teledyne V.: *Grasshopper3 USB3*. https://www.flir.de/products/grasshopper3-usb3/?model=GS3-U3-123S6M-C&vertical=machine+vision&segment=iis. – Zugriffsdatum: 27.05.2024

[84] Statista: *Marktanteile der führenden Betriebssysteme weltweit von Januar 2009 bis März 2024*. https://de.statista.com/statistik/daten/studie/157902/umfrage/marktanteil-der-genutzten-betriebssysteme-weltweit-seit-2009/. – Zugriffsdatum: 15.05.2024

[85] StereoLabs: *What is the camera focal length and field of view?* https://support.stereolabs.com/hc/en-us/articles/360007395634-What-is-the-camera-focal-length-and-field-of-view. – Zugriffsdatum: 13.11.2024

[86] StereoLabs: *ZED Mini Stereo Camera*. https://www.stereolabs.com/en-de/store/products/zed-mini. – Zugriffsdatum: 16.10.2024

[87] StereoLabs: *ZED Mini Stereo Camera*. https://cdn.sanity.io/files/s18ewfw4/staging/ebcd46896092d1ee6212b7f4d81aaa1c479c2440.pdf/ZED%20Mini%20Datasheet%20v1.1.pdf. – Zugriffsdatum: 16.10.2024

[88] Turk, Greg ; Levoy, Marc: Zippered polygon meshes from range images. In: *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques*. New York, NY, USA : Association for Computing Machinery, 1994 (SIGGRAPH '94), S. 311–318. – URL https://doi.org/10.1145/192161.192241. – ISBN 0897916670

[89] Tuytelaars, Tinne ; Gool, Luc V.: *SURF : Speeded Up Robust Features Herbert Bay*. 2006. – URL https://api.semanticscholar.org/CorpusID:267878288

[90] UltiMaker: *What accuracy can be achieved with an UltiMaker S series printer?* https://support.ultimaker.com/s/article/000002952#:~:text=The%20UltiMaker%20S5%20and%20S7,of%20less%20than%200.2%20mm.#. – Zugriffsdatum: 14.05.2024

[91] Vlaminck, Michiel ; Luong, Hiep ; Philips, Wilfried: Multi-resolution ICP for the efficient registration of point clouds based on octrees. In: *2017 Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, 2017, S. 334–337

[92] Wang, R. ; Law, A.C. ; Garcia, D. et a.: Development of structured light 3D-scanner with high spatial resolution and its applications for additive manufacturing quality assurance. In: *The International Journal of Advanced Manufacturing Technology* 117 (2021), August, S. 845–862

[93] Weber, Christopher ; Hahmann, Stefanie: Methods for Feature Detection in Point Clouds. In: *OpenAccess Series in Informatics* 19 (2010), 03, S. 90–99

[94] Weckenmann, Albert: *Koordinatenmesstechnik : flexible Strategien für funktions- und fertigungsgerechtes Prüfen*. Carl Hanser Verlag GmbH & Company KG, 2012. – URL https://www.google.de/books/edition/Koordinatenmesstechnik/lqpPAgAAQBAJ?hl=de&gbpv=0. – ISBN 9783446429475

[95] Winkelbach, Simon ; Molkenstruck, Sven ; Wahl, Friedrich: Low-Cost Laser Range Scanner and Fast Surface Registration Approach, 09 2006, S. 718–728. – ISBN 978-3-540-44412-1

A Anhang

A.1 Herleitung Stereophotogrammetrie

Nach dem Strahlensatz erhält man:

$$\frac{B_1}{b} = \frac{\frac{a}{2} - G}{g} \tag{A.1}$$

$$\frac{-B_2}{b} = \frac{\frac{a}{2} + G}{g} \tag{A.2}$$

Beide Funktionen werden nach G umgestellt.

$$\frac{B_1 \cdot g}{b} = \frac{a}{2} - G \rightarrow \frac{B_1 \cdot g}{b} - \frac{a}{2} = -G \tag{A.3}$$

$$\frac{-B_2 \cdot g}{b} = \frac{a}{2} + G \rightarrow \frac{-B_2 \cdot g}{b} - \frac{a}{2} = G \tag{A.4}$$

Beide Funktionen werden gleichgesetzt und nach g umgestellt.

$$\frac{B_1 \cdot g}{b} - \frac{a}{2} = \frac{B_2 \cdot g}{b} + \frac{a}{2} \tag{A.5}$$

$$\frac{B_1 \cdot g}{b} = \frac{B_2 \cdot g}{b} + a \tag{A.6}$$

$$B_1 \cdot g = B_2 \cdot g + a \cdot b \tag{A.7}$$

$$B_1 \cdot g - B_2 \cdot g = a \cdot b \tag{A.8}$$

$$g \cdot (B_1 - B_2) = a \cdot b \tag{A.9}$$

$$g = \frac{a \cdot b}{B_1 - B_2} \tag{A.10}$$

Mit der Definition der Parallaxe $p = B_1 - B_2$ gilt:

$$g = \frac{a \cdot b}{p}$$

Erklärung zur selbständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original