Hochschule für Technik und Wirtschaft Berlin
University of Applied Sciences

Angewandte Informatik

Masterarbeit

Handcontroller für die Microsoft HoloLens

Vorgelegt an der Hochschule für Technik und Wirtschaft Berlin

bei Prof. Dr.-Ing. Thomas Jung und Robert Meyer

zur Erlangung des akademischen Grades Master of Science (M.Sc.)

Vorgelegt von Julian Löhr

Eingereicht am 20.10.2017

Ich versichere, dass ich die Kapitel der Arbeit, für die ich als Verfasser genannt werde, selbständig verfasst habe, dass ich keine anderen, als die angegebenen Quellen und Hilfsmittel benutzt habe und dass ich diese Arbeit bei keinem anderen Prüfungsverfahren vorgelegt habe.

Ort, Datum Unterschrift

This Master‘s thesis is about the development of a hand controller for the Microsoft HoloLens device. The main criteria for such a hand controller are no direct line of sight and no use of external base stations. As no current Virtual Reality tracking system meets the criteria, a novel tracking system based on SLAM devices must be developed.

The SLAM based tracking system uses SLAM devices. Each device tracks its pose, as well as its local map. In an initial phase the local maps of each device are aligned to create a shared reference frame. After that the resulting transformation can be used to convert the pose from one local reference frame to the other.

For the map alignment process a transformation estimation is made, to make sure it aligns correctly and to speed up the process. For the estimation the compass of each device shall be used to compute their relative heading. As applications can’t access the compass sensor on the HoloLens, some alternatives are developed. One is the use of a visual alignment process involving the Vuforia platform. Another is the assumption, that the user holds the controller device in its hand and gazes at it while starting it.

As the hand controller device, a Lenovo Phab 2 Pro is used. As it is a Google Tango device, it features all the necessary functions, like pose tracking and creating a local map. For the map alignment process different variations of the iterative closet points algorithm are used and tested.

After an evaluation the concept proves to meet the criteria. However, it lacks precision. The time limit for the initial phase can also only be met with the use of the assumption based transformation estimation. Only using the heading, providing only a rotation estimation, results in misalignment and takes too long. Additionally the Google Tango platform lacks robustness, accuracy and a relocation as well as proper drift correction mechanisms.

Inhaltsverzeichnis

1. Motivation

Die HoloLens ist eine neue Mixed Reality Brille von Microsoft. Das Gerät ist ein eigenständiger Computer und verfügt über eine Vielzahl an Sensoren. Zum Interagieren werden verschiedene Möglichkeiten geboten, wie Gaze, Gesten oder Sprachbefehle. Bei dieser ersten Generation von neuen Mixed und Augmented Reality Brillen ist die Standardeingabeinteraktion der Gaze im Verbund mit einer Tap Geste. Ähnlich wie an einem herkömmlichen Desktop Computer wird per Gaze ein Cursor gesteuert und per Tap eine Klick Aktion ausgeführt. Diese Art der Interaktion ist aus verschiedenen Gründen suboptimal.

Der Nutzer muss ständig den Kopf drehen und bewegen, was bei längerem Gebrauch zu Beschwerden führen kann. Hinzu kommt, dass beim Ausführen der Tap Geste der Kopf und somit der Gaze stillgehalten werden muss, da der Nutzer ansonsten daneben klicken könnte. Dies gestaltet sich besonders bei kleineren Elementen schwierig, da durch die physiologische Ausführung der Tap Geste, der Nutzer auch den Kopf leicht bewegt. Außerdem muss sich die Hand mitausgestrecktem Arm beim Ausführen von Gesten vor der HoloLens befinden. Beim kontinuierlichen Gebrauch der HoloLens und der Tap Geste ermüdet der Arm des Nutzers relativ schnell. Dies verhindert eine längerfristige Nutzung der HoloLens und dessen Gestensteuerung. Weiterhin ist es für viele Nutzer entgegen ihrer Intuition, dass die Klick Aktion nicht an der Position der Hand ausgeführt wird, sondern an der Position des Gaze Cursors, was zunächst für Verwirrung sorgt.

Aus diesen Gründen müssen weitere und alternative Eingabe Methoden für die HoloLens entwickelt werden. Im Virtual Reality Bereich werden die Handpositionen der Nutzer mittels zwei Handcontrollern getrackt. Diese Controller ermöglichen die Ermittlung von Position und Ausrichtung und bieten zudem die Möglichkeit von weiteren Eingaben über Knöpfe, analog Sticks oder Touch Pads. In den Anwendungen werden die Controller entweder als virtuelle Hände oder als eine Art Laserpointer umgesetzt. Letztere fungieren wie eine drei dimensionale Computermaus und ermöglichen es dem Nutzer auch mit weiter entfernten Objekten intuitiv und genau zu interagieren. Daher soll nun auch für die HoloLens ein Handcontroller entwickelt werden, um ähnlich intuitive Interaktionen zu ermöglichen.

2. Ziele und Anforderungen

Das Ziel dieser Arbeit ist einen Handcontroller für die HoloLens zu entwickeln. Dieser soll primär eine Alternative zur Hauptinteraktion per Gaze und Tap Geste darstellen, indem er wie bei gängigen Virtual Reality Systemen als eine Art Laserpointer oder drei dimensionale Computermaus fungiert. Die Möglichkeit das Verfahren als allgemeines Tracking System für Zubehör und Weiteres im Verbund mit der HoloLens zu nutzen, soll eine untergeordnete Rolle spielen.

Das Nutzungsverhalten des Controllers soll sich an dem der HoloLens orientieren. Daraus resultieren die folgenden Anforderungen an den zu entwickelnden Controller. Ähnlich der HoloLens soll der Handcontroller ohne größere Kalibrierungsphase, aber evtl. mit einer kurzen Initialisierung, unmittelbar einsatzbereit sein. Daher soll die Zeitspanne zum Initialisieren und Kalibrieren, bis der Controller vollständig einsatzbereit ist, maximal 20 Sekunden betragen. Die HoloLens verfügt über ein Inside-Out Tracking, welches nicht auf externe Stationen oder explizite Referenzpunkte angewiesen ist. Dadurch lässt sich diese ohne vorherigen Aufbau in jedem beliebigen Ort nutzen. Diese Eigenschaft soll auch beim Handcontroller erhalten bleiben, sodass dieser ein eigenständiges Gerät sein soll, welches Inside-Out Tracking ohne externe Referenzstationen betreibt. Außerdem sollen keine Umfangreichen Modifikationen an der HoloLens vorgenommen werden. Dies dient dazu, dass das Tracking Verfahren einfach portierbar und ohne großen Aufwand mit weiteren HoloLens kompatibel ist. Weiterhin soll auf ein System verzichtet werden, welches eine direkte Sichtverbindung zwischen HoloLens und Controller voraussetzt, da diese nicht in allen Nutzungsszenarien sichergestellt werden kann. Der Handcontroller soll wie die HoloLens nur für Innenbereiche ausgelegt sein (Microsoft Corporation 2017). Dabei sollen die Hauptumgebungen zur Nutzung Wohn- und Büroräume sein. Ein stabiles Tracking muss in statischen Umgebungen sichergestellt sein, dynamische Umgebungen müssen wie bei der HoloLens zunächst nicht unterstützt werden (Microsoft Corporation 2017). Die Genauigkeit des Handcontrollers soll höchstens 0,5 Centimeter Abweichung bei der Position betragen. Hologramme der HoloLens sind in einem Raum von 5 Metern stabil (Microsoft Corporation 2017). Daher soll die Orientierungsabweichung maximal 0,11 Grad betragen, was 1 Centimeter Abweichung auf 5 Meter Entfernung entspricht.

3. Aufbau der Arbeit

Nachdem in Kapitel 2 die Ziele und Anforderungen definiert werden, werden in Kapitel 4 zunächst die gängigen Handcontroller und Tracking Systeme vorgestellt. Diese werden anschließend in Kapitel 5 kurz den Anforderungen gegenübergestellt, um das Kernthema der Arbeit vorzustellen. In Kapitel 7 wird das Kernthema, ein neuartiges Tracking System für HoloLens und Handcontroller, erarbeitet und im Detail beschrieben. Weiterhin wird in Kapitel 8 die zu verwendende Hardware für den Handkontroller festgelegt. Nach einer Implementation des Trackings System in Kapitel 9, wird dieses zusammen mit dem Handkontroller in Kapitel 10 getestet und ausgewertet. Abschließend wird in Kapitel 11 ein Fazit gezogen und ein Ausblick vorgestellt.

4. Stand der Technik

4.1 SteamVR Tracking

SteamVR Tracking, auch Lighthouse genannt, ist ein von Valve entwickeltes Tracking System. Es wird primär von der Virtual Reality Brille HTC Vive genutzt, ist nun jedoch auch für weitere Lizenznehmer verfügbar (Valve Corporation 2017). Es ist für Innenräume konzipiert und spannt einen bis zu 5 Meter großen Tracking Raum auf (Valve Corporation 2017). Das System wird als Inside-Out Tracking kategorisiert, was bedeutet, dass die Geräte ihre Position eigenständig anhand von externen Referenzpunkten ermitteln.

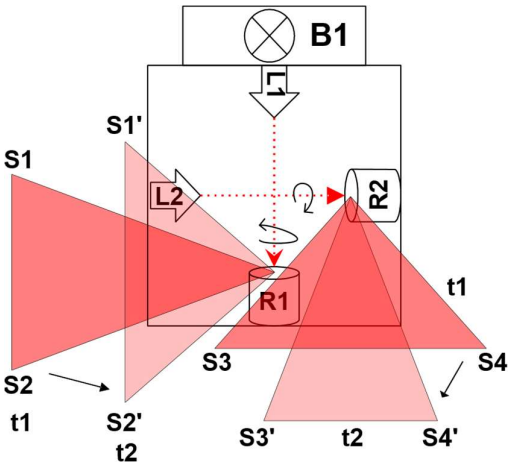


Abbildung 4-1: Aufbau Basisstation (Islam, et al. 2016)

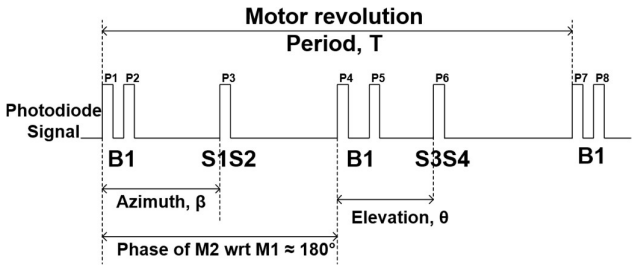


Abbildung 4-2: Winkelermittlung per Lichtimpulse (Islam, et al. 2016)

Kernelement des Systems sind bis zu zwei Basisstationen (s. Abbildung 4-1), welche jeweils mit einem Infrarot Lichtblitz $B1$ und zwei orthogonal zueinander ausgerichteten sich rotierenden Linienlasern $L1$-$R1$ und $L2$-$R2$ ausgestattet sind. Die zu trackenden Geräte sind mit mehreren in einem festen und bekannten Muster angeordneten Infrarot Sensoren ausgestattet. Die Position und Rotation der Geräte wird über Trigonometrie ermittelt. Dazu sendet eine Basisstation zunächst einen Lichtblitz $B1$ aus. Dieser signalisiert den Start einer Rotationsphase. Daraufhin durchfährt einer der Linienlaser $S1$-$S2$ den Raum. Jeder Infrarot Sensoren auf einem Gerät misst die Zeitdifferenz zwischen Lichtblitz und rotierendem Laser. Da die Laser mit bekannter und fester Geschwindigkeit rotieren, lässt sich mithilfe der Zeitdifferenz und Geschwindigkeit der Winkel des Sensors zur Basisstation ermitteln (s. Abbildung 4-2). Anschließend signalisiert ein weiterer Lichtblitz $B1$ das durchfahren des anderen Rotationslasers $S3$-$S4$. Somit sind für jeden Sensor die Winkel ihrer Projektionen in zwei Achsenebenen des drei dimensionalen Raumes bekannt (s. Abbildung 4-3). Da auch die genaue Anordnung und der Abstand der Sensoren bekannt ist, lässt sich mit einem Linearen Gleichungssystem die genaue Position im Raum bestimmen. (Islam, et al. 2016, Valve Corporation 2017)

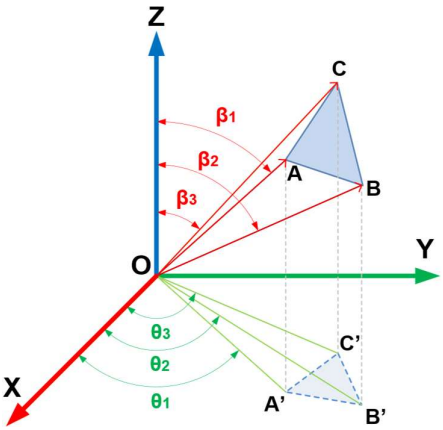


Abbildung 4-3: Projektion der Sensoren auf zwei Achsenebenen (Islam, et al. 2016)

Laut Hersteller ist die Genauigkeit des Systems Millimeter genau (Valve Corporation 2017). Der Vorteil dieses Systems ist, dass die Basisstation keine weitere Kommunikation oder Synchronisation mit den Geräten oder Software benötigt. Somit kann ein Aufbau der Basisstationen theoretisch unendliche viele Geräte unterstützen. Um die Genauigkeit weiter zu erhöhen und die Latenz zu verringern, ist es zudem möglich einen Controller mit einer inertialen Messeinheit (IMU) auszustatten (Valve Corporation 2017).

4.2 PlayStation Move & VR aim controller

Der PlayStation Move Handcontroller wurde von Sony Interactive Entertainment als Zubehör für die PlayStation 3 und PlayStation 4 entwickelt. Im Rahmen der neuen PlayStation VR Brille wurde außerdem ein VR aim controller entwickelt, welcher auf dem gleichen Verfahren basiert. Das System wird als Outside-In Tracking kategorisiert, da eine externe feste Kamera passive Elemente an den zu trackenden Geräten wahrnimmt. Die verwendete Kamera zum Tracken ist eine Stereo Tiefenkamera und wird PlayStation Camera genannt. Die in Abbildung 4-4 gezeigten Handcontroller besitzen eine farbliche Kugel fester Größe. Über die Position der Kugel im Kamerabild lässt sich dessen Position vor der Kamera herleiten. Im ursprünglichen Verfahren wurde die Entfernung des Controllers über die Größe der Kugel im Kamerabild bestimmt. Um die Genauigkeit zu erhöhen nutzt die PlayStation Camera der PlayStation 4 eine Stereo Tiefenkamera (Peckham 2016). Die Controller sind ebenfalls mit IMUs ausgestattet, um die Orientierung und

Abbildung 4-4: PlayStation Move (links) & PlayStation VR aim controller (rechts) (Sony Interactive Entertainment Europe Limited 2017)

Drehbewegungen zu erfassen.

4.3 Oculus Touch

Für die Oculus Rift Virtual Reality Brille wurde von Oculus VR der Handcontroller namens Oculus Touch entwickelt. Für das Tracking wird ein auf Infrarot basierendes Outside-In System genutzt. Die Geräte besitzen eine Infrarot durchlässige Abdeckung (Goradia, Doshi and Kurup 2014). Darunter befinden sich mehrere Infrarot LEDs in einer bestimmen Konstellation. Der stationäre Oculus-Sensor besteht aus einer Infrarot Kamera. Über die Größe und Position der Konstellation im Kamerabild, lässt sich die genaue Position und Rotation im Raum bestimmen. Derzeit wird nur die Verwendung von einem Sensor unterstützt. Das Verwenden von mehreren Sensoren für ein 360 Grad Tracking gilt lediglich als experimentell (Oculus VR, LLC 2017).

4.4 Microsoft Motion Controllers

Für die neuen Windows Mixed Reality Immersive Headsets hat Microsoft die Motion Controller entwickelt. Diese neuen Headsets nutzen selbst ein Inside-Out Tracking, welche auf der HoloLens Technologie basiert und ohne externe Sensoren oder Stationen auskommt (Microsoft Corporation 2017). Das Tracking der Motion Controller hingegen ist ein Kamerabasiertes Outside-In Tracking. Wie in Abbildung 4-5 zu sehen, befinden sich am Headset zwei Kamera Sensoren. Die Controller besitzen einen Ring, welcher mit mehreren LEDs in bestimmter Konstellation bestückt ist (s. Abbildung 4-6). Die Kameras am Headset erkennen die Konstellation am Controller und können drüber die relative Position des Controllers zum Headset feststellen. Der Vorteil dieses Systems ist, dass trotz Outside-In Trackings keine externe feste Referenzstation benötigt wird. Die Controller müssen sich dafür jedoch ständig im Sichtbereich der Sensoren vor dem Headset befinden. Somit ergibt sich ein eingeschränkter Trackingbereich für die Controller. Um die Genauigkeit zu erhöhen, verfügen die Handcontroller zudem über IMUs. Mit diesen lässt sich außerdem kurzfristig das Tracking überbrücken, sollte ein Controller den Sichtbereich der Kameras verlassen. Die Genauigkeit nimmt jedoch rapide ab, sollte sich er Controller längerfristig außerhalb des Trackingbereiches aufhalten. Beim Drehen des Kopfes kann es zudem zu Sprüngen keim Tracking des Controllers kommen (Lang 2017).

Abbildung 4-5: Acer Windows Mixed Reality Headset (Jackson 2017)

Abbildung 4-6: Microsoft Motion Controllers (Microsoft Corporation 2017)

Die Microsoft Motion Controller sind jedoch inkompatibel mit der HoloLens (Microsoft Corporation 2017). Es gibt bereits experimentelle Projekte, um ein ähnliches Verfahren für HoloLens umzusetzen. Afternow nutzt in ihrer Demo "Extend Your Phone Screen with the HoloLens" ein Handy als Handcontroller. Auf dem Display des Handys wird ein Bild angezeigt, welches als Bilder Marker für die Vuforia Plattform fungiert. Über die Webcam der HoloLens wird das Bild auf dem Display erkannt und zur Berechnung der relativen Position des Handys zur HoloLens genutzt (Vander Does, Question about your turn your phone into a HoloLens controller PoC 2017). Bei dieser Variante wird weiterhin das Touch Display des Handys als Eingabeelement für den Controller genutzt (Vander Does, Extend Your Phone Screen with the HoloLens 2017).

4.5 Razer Hydra & STEM System

Die Razer Hydra Gamecontroller und das STEM System wurden beide von Sixsense entwickelt und basieren auf der gleichen Technologie. Während die Razer Hydra Controller für die Bewegungssteuerung am Desktop Computer entwickelt wurde, wurde mit dem STEM System ein Nachfolgesystem für Virtual Reality Anwendungen geschaffen. Zum Einsatz kommt ein Inside-Out Tracking auf Basis von Magnetfeldern. In einer Basisstation ist eine auf Magnetspule verbaut. Diese generiert ein Wechselfeld, welches wiederum von bis zu 5 Controllern genutzt wird, um deren Position und Orientierung zu ermitteln. Die Genauigkeit wird bei der Razer Hydra mit einem Millimeter und einem Grad angegeben (Razer Inc. 2017). Die Reichweite beträgt beim STEM System 2,5 Metern um die Basisstation. Der Vorteil dieser Technologie gegenüber optischen Verfahren ist, dass es keine Sichtverbindung benötigt und somit unempfindlich gegenüber Verdeckung ist (Sixense Entertainment 2014).

5. Abgrenzung der Aufgabe

Die gängigen Six Degree of Freedom (6DOF) Handcontroller Systeme wurden in Kapitel 4 vorgestellt. Jedoch erfüllt keines der Systeme die Anforderungen zufriedenstellend. Alle Systeme bis auf die Microsoft Motion Controller benötigen externe und teilweise fixe Basisstationen. Diese können somit ausgeschlossen werden, da der Handcontroller wie die HoloLens ohne externe Basisstation funktionieren soll. Weiterhin basieren alle Systeme, bis auf Razer Hydra und das STEM System, auf optischen Methoden zum Tracking. Dafür wird eine direkte Sichtlinie vorausgesetzt, welche nicht sichergestellt werden kann und somit diese Systeme zunächst nicht weiter betrachtet werden sollen. Zusätzlich besitzt die HoloLens lediglich eine Webcam mit beschränktem Sichtfeld, wodurch ein visuelles Tracking, wie es für die Microsoft Motion Controller eingesetzt wird, nur in einem eingeschränkten Raum verfügbar wäre. Die vier Umgebungskameras der HoloLens stehen einer Anwendung nicht zur Verfügung. Dies ist vermutlich der Grund, warum die Microsoft Motion Controller nicht mit der HoloLens kompatibel sind. Das STEM System erfüllt insofern die Anforderungen, dass es keine direkte Sichtverbindung benötigt und die Basisstation nicht fix sein muss. Jedoch verfügt die HoloLens nicht über die entsprechende Technik und müsste modifiziert werden. Dies soll ebenfalls möglichst verhindert werden und wird somit ausgeschlossen. Zudem ist die Reichweite auf 2,5 Meter von der Basisstation begrenzt. Dies ist zwar innerhalb der Armlänge eines Menschen, schränkt aber das Tracking System insofern ein, dass es nicht für weiter entfernte unabhängige Geräte genutzt werden kann.

Da keiner der Handcontroller die Anforderungen erfüllt, muss eine Eigenentwicklung durchgeführt werden. Die Entwicklung kann zunächst in drei Teilaufgaben unterteilt werden. Zunächst muss ein geeignetes Tracking Verfahren erarbeitet werden, sodass kontinuierlich die relative Position von Handcontroller zu HoloLens erfasst wird. Weiterhin muss unter anderem auf Basis des Tracking Systems die Kontroller Ergonomie erarbeitet werden. Dies beinhaltet sowohl die Anzahl, Art und Anordnung der Eingabeelemente und Sensoren, als auch die Form und das Aussehen des Handcontrollers. Zudem muss ein geeignetes Kommunikationsmedium gewählt werden, um die entsprechenden Informationen zwischen HoloLens und Handcontroller austauschen zu können.

Da wie bereits beschrieben keines der gängigen Tracking Systeme die Anforderungen erfüllt, muss ein anderer neuartiger Ansatz gewählt werden. Um den Handcontroller in allen Umgebungen zu nutzen, soll dieser wie die HoloLens seine Position und Orientierung eigenständig im Raum erfassen können. Wenn beide Geräte unabhängig ihre Position im Raum bestimmen, muss lediglich ein gemeinsamer Referenzrahmen geschaffen werden. Überträgt man nun jeweils die beiden Positionen in den gemeinsamen Referenzrahmen, kann die relative Position und Ausrichtung zueinander bestimmt werden. Zur Bestimmung des Referenzrahmens, soll der Controller ebenfalls wie die HoloLens die Umgebung erfassen. Anschließend sollen die lokalen Referenzrahmen über die erfasste Umgebung zu einander ausgerichtet werden, um so einen globalen Referenzrahmen zu schaffen. Der Vorteil eines solchen Verfahren ist, sofern der Handcontroller über die entsprechende Technik verfügt, keine weitere Technik oder Modifikation der HoloLens benötigt

wird und das Tracking per Software gelöst werden kann. Weiterhin wäre die Reichweite zwischen HoloLens und Handcontroller nach dem Herstellen des gemeinsamen Referenzrahmens unbegrenzt.

Da das Tracking Verfahren neuartig ist, wird zunächst lediglich ein Prototyp entwickelt. Die Teilaufgabe der Ergonomie wird daher zunächst nicht weiter betrachtet. Der Kontroller muss lediglich über die benötigten Sensoren verfügen und zusätzlich mindestens ein Eingabeelement besitzen, um die Tap Aktion auszulösen. Selbsterklärend ist natürlich, dass der der Controller in der Hand gehalten werden kann. Außerdem wird die Teilaufgabe der Kommunikation als bereits gelöst angesehen. Es gibt mehrere gängige und etablierte Verfahren zu Kommunikation zwischen Geräten. Für die kabellose Kommunikation wird häufig WLAN, Bluetooth oder wie bei den Xbox Controllern ein ebenfalls auf dem 2,4 Ghz Band basierendes proprietäres Protokoll eingesetzt. Da die HoloLens sowohl WLAN als auch Bluetooth unterstützt, kann daher für die Implementierung frei zwischen diesen gewählt werden. Somit verbleibt lediglich die Teilaufgabe des Tracking Verfahrens, welche es in dieser Arbeit zu lösen gilt.

6. Grundlagen

6.1 3D Daten

6.1.1 Punktwolken

Eine Punktwolke ist eine endliche Menge von Punkten, auch Vertices genannt, in einem drei dimensionalen kartesischen Raum. Somit gilt für eine Punktwolke

$$P = \{p_i : i = 1, \dots, n;\ p_i \in \mathbb{R}^3\} \ .$$

Ein Punkt kann neben seiner Position auch aus weiteren Informationen bestehen, wie zum Beispiel seiner Farbe oder einem dazugehörigen Normalenvektor (Rothmaier 2012, 19). Um einen Punkt im Raum zu verändern, kann dieser transformiert werden. Möchte man eine Punktwolke transformieren, so wendet man die Transformation einfach auf alle ihre Punkte an. Die drei typischen Transformationen sind die Translation

$$x' = x + t \ ,$$

welche eine Punktwolke bzw. den Ursprung verschiebt, die Rotation

$$x' = R * x \ ,$$

um eine Punktwolke um den Ursprung zu rotieren, und die Skalierung

$$x' = S * x \ ,$$

welche den Abstand zwischen den Punkten verändert. Die Translation und die Rotation erhalten die Winkel und Abstände von Geraden und Punkten. Zusammen werden diese beiden auch als euklidische Transformation bezeichnet. Alle drei Transformationen gehören zu den affinen Transformationen, diese erhalten Parallelen und Abstandsverhältnisse. (Süße and Rodner 2014, 21 f., 261 f.)

Der Unterschied zwischen Punktwolken und herkömmlichen Polygonmodellen ist, dass Punktwolken lediglich diskrete unabhängige Punkte darstellen und keine geschlossene Oberfläche. Bei der Aufnahme von Punktwolken, wird eine Abtastung durchgeführt, welche in diskreten Punkten resultiert (Linsen 2001, Berger, et al. 2014). Die Rekonstruktion einer Oberfläche aus einer Abtastung wird in Kapitel 6.1.5 erläutert.

6.1.2 Aufnahmesysteme

Für die Aufnahme von 3D Daten gibt es drei verschiedene Konzepte. Diese sind Triangulation, time-of-flight und ein einfaches Bild mit Hintergrundwissen. Letzteres ist noch nicht weit verbreitet und wird deshalb nicht näher beleuchtet. Kameras, welche 3D Daten aufnehmen, werden 3D-Kameras oder auch Tiefenkameras genannt. (Nüchter 2009, 9)

Das time-of-flight Konzept misst die Zeit t zwischen dem Aussenden eines Signals und dem Empfangen des reflektierten Signals. Sender und Empfänger sind in der Regel direkt nebeneinaner positioniert, sodass die Distanz von Sender zu Objekt gleich ist wie Objekt zu Empfänger. Außerdem muss die Geschwindigkeit v des Signals im Medium, meist Luft, bekannt sein. Dann kann mittels

$$D = v\frac{t}{2}$$

die Distanz D zum Objekt berechnet werden. Als Signal wird meist Licht oder Schall genutzt. Da die Geschwindigkeit von Licht sehr hoch ist, muss bei auf Licht basierten 3D-Kameras der Zeitmesser eine Auflösung von Pikosekunden haben. Eine Sonderform der time-of-flight Distanzmessung ist die Messung per Phasenverschiebung. Dabei wird ein kontinuierliches Signal ausgesandt. Die Amplitude des Signals wird mit der Sinusfrequenz f moduliert. Beim Empfangen wird die Phasendifferenz $\Delta\varphi$ zwischen dem empfangenen Signal und dem derzeitig ausgesandten Signal festgestellt. Mittels

$$D = \frac{\Delta\varphi\ v}{4\pi f}$$

wird die Distanz zum Objekt berechnet. Da die Phasendifferenz proportional zur Wellenlänge des modulierten Signals ist, kann meist nicht die gesamte Distanz mit einer Modulation gemessen werden. Daher werden häufig zwei Modulationen eingesetzt. Eine höher frequentierte Modulation für eine genaue Auflösung und eine niedrigere Frequenz für die grobe Distanz. (Nüchter 2009, 9 f.)

Bei der Triangulation wird eine Lochkamera genutzt. Ein Sender sendet ein Signal aus, welches vom Objekt reflektiert wird. Eine Lochkamera mit Abstand *L* zum Sender empfängt dieses Signal (s. Abbildung 6-1). Über die Abweichung x zur Bildmitte kann die Distanz mittels

$$D = f\frac{L}{x}$$

berechnet werden. Dabei ist *f* die Bildweite der Lochkamera. Stereo Kameras nutzen ebenfalls dieses Prinzip, wenn auch in leicht abgewandelter Form. Statt eines Emitters wird eine zweite Kamera genutzt (s. Abbildung 6-2). Über die perspektivische Differenz der beiden Bilder kann die Tiefe eines Bildpunktes berechnet werden. Die Formel muss wie folgt angepasst werden

$$D = f\frac{L}{x_l - x_r}\ ,$$

dabei sind x_l und x_r die Abweichungen der linken und rechten Kamera. (Nüchter 2009, 19 f.)

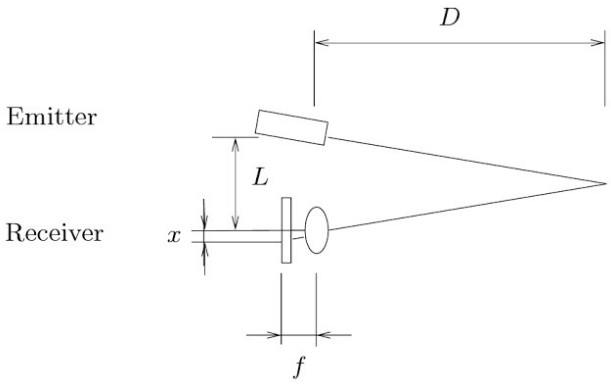


Abbildung 6-1: Triangulation (Nüchter 2009, 11)

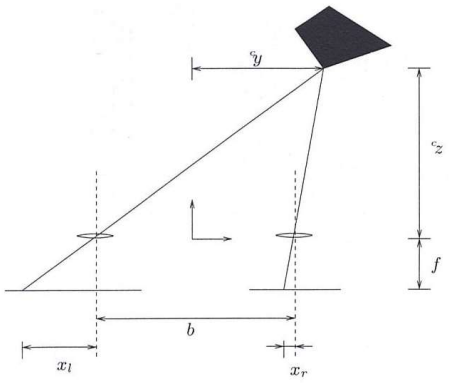


Abbildung 6-2: Stereo Kamera (Nüchter 2009, 21)

6.1.3 Dichte

„Bei der Dichte [einer Punktwolke] handelt es sich um die Punkte pro Volumeneinheit, welche sich aus der Verteilung der Punktmenge ableitet" (Rothmaier 2012, 20). Bei der Aufnahme von 3D Daten kann eine zu hohe Dichte entstehen, welche sich als Problem bei der Verarbeitung darstellt (Lee, Woo und Suk 2001, Rothmaier 2012, 26). Zudem sind Aufnahmen häufig verrauscht (Orts-Escolano, et al. 2013). Um die Dichte zu verringern oder möglichst viel Rauschen zu entfernen kann der sogenannte Voxelgitter Filter genutzt werden (Orts-Escolano, et al. 2013, Rothmaier 2012, 26 f.). Der Vorgang die Anzahl der Punkte in einer Punktwolke zu verringern wird Downsampling genannt. Bei dem Voxelgitter Filter wird der drei dimensionale Raum in gleich große Würfel, sogenannte Voxel, unterteilt. Für jeden dieser Würfel wir der geometrische Schwerpunkt seiner in ihm befindlichen Punkte berechnet. Dieser berechnete Schwerpunkt ersetzt anschließend die in ihm befindlichen Punkte. Über die Größe der verwendeten Würfel lässt sich die neue Dichte festlegen (Rothmaier 2012, 26 f.).

6.1.4 Point Set Registration

Besitzt man zwei Punktwolken und möchte diese so ausrichten, dass sie ganz oder teilweise übereinanderliegen, spricht man von Point Set Registration oder auch Point Matching. Häufige Anwendungsfälle sind zwei konsekutive Aufnahmen, welche zu einer Gesamtaufnahme zusammengefügt werden soll, oder eine Detailaufnahme von einem Objekt, welches es in einer Gesamtaufnahme zu lokalisieren gilt (Besl und McKay 1992, Nüchter 2009, 35, Zhang 1994, Chen und Medioni 1992). Die Point Set Registration wird als Optimierungsproblem beschrieben, bei dem eine Transformation gesucht wird, welche eine Kostenfunktion minimiert (Nüchter 2009, 35). Die Kostenfunktion ist dabei die Entfernung zwischen den beiden Punktwolken (Besl und McKay 1992, Chen und Medioni 1992, Zhang 1994).

Bei der Point Set Registration unterscheidet man zwischen starren und elastischen Point Matching. Die starre Point Set Registration sucht lediglich nach einer euklidischen Transformation, bestehend aus Translation und Rotation. Beim elastischen Point Matching werden je nach Anwendungsfall affine oder auch nicht affine Transformation, wie zum Beispiel Verzerrung, in den Suchraum aufgenommen. Dadurch gestaltet sich die elastische Point Set Registration als komplexer und schwieriger. (Süße and Rodner 2014, 557 ff.)

Für die Point Set Registration gibt es eine Vielzahl an Algorithmen für verschiedene Anwendungsfälle. Der weit verbreitete Iterative Closest Point Algorithmus wird in Kapitel 6.2 näher erläutert.

6.1.5 Rekonstruktion

Rekonstruktion bezeichnet das Verfahren aus einer Punktwolke eine digitale Oberflächenrepräsentation zu erstellen. Bei der Aufnahme von Punktwolken, wird lediglich eine diskrete Abtastung durchgeführt. In vielen Anwendungsfällen muss man diese diskrete Abtastung in eine vollständige digitale Repräsentation umwandeln. Aufnahmen weisen zudem häufig verschiedene Defekte auf, wie Rauschen, ungleichmäßige Dichte, Versatz zwischen Aufnahmen oder schlicht fehlende Daten, die es zu beheben gilt. Es wurde eine Vielzahl von verschiedenen Ansätzen und Verfahren entwickelt, um eine Rekonstruktion durchzuführen. Dabei ist wichtig zu beachten, dass es kein allgemeines Verfahren für alle Anwendungsfälle geben kann. Jedes Verfahren basiert auf bestimmten Annahmen über das zu rekonstruierente Objekt und seine Defekte. Somit muss für jeden Anwendungsfall individuell das am besten passende Verfahren ausgewählt werden, welches die korrekten Annahmen beim Rekonstruieren macht. (Berger, et al. 2014)

Eine verbreitete Methode ist die des Tuncted Signed Distance Field (TSDF). Bei diesem Verfahren wird der Raum in Voxel unterteilt. Jeder Voxel beinhaltet einen einzelnen Skalaren Wert, welcher die Distanz zur am nächsten Oberfläche beschreibt. Positive Werte bedeuten, dass der Voxel sich außerhalb eines Objektes befindet, Negative, dass sich der Voxel innerhalb befindet und ein Wert von Null, dass die Oberfläche durch diesen Voxel führt. Bei jeder Aufnahme eines Tiefenbildes wird pro Pixel ein Raycast ausgeführt. Jeder Voxel den dieser Raycast durchläuft wird entsprechend des Tiefenwertes des Pixels gewichtet aktualisiert. Um Interferenz mit Rückseiten oder verdeckten Oberflächen zu vermeiden, sollte die Gewichtung nach der gemessenen Pixeltiefe rapide abnehmen (Curless and Levoy 1996). Je mehr Aufnahmen getätigt werden desto genauer wird das TSDF und Rauschen wird eliminiert. Um ein Polygonmodell aus dem TSDF zu extrahieren können verschiedene Verfahren, wie Marching Cubes oder Raycasting, genutzt werden. Die Isofläche des Objektes wird wie in Abbildung 6-3 durch Voxel mit dem Wert Null repräsentiert. (Klingensmith, et al. 2015, Dryanovski 2016)

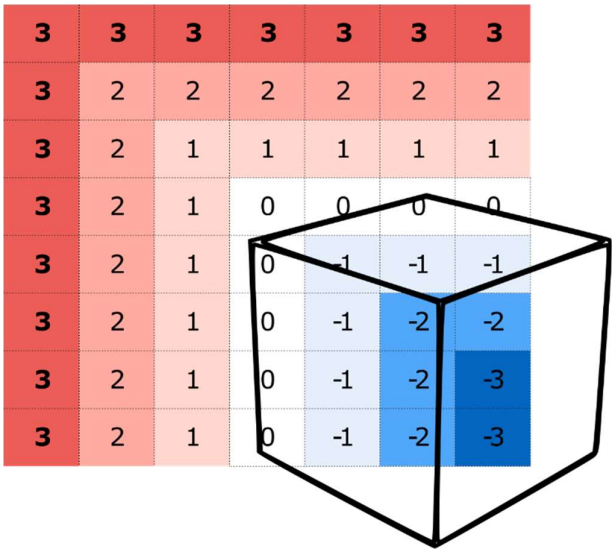


Abbildung 6-3: TSDF Gitter mit Isofläche (Dryanovski 2016)

6.2 Iterative Closest Points

6.2.1 Grundalgorithmus

Der Iterative Closest Points (ICP) Algorithmus wurde etwa zur gleichen Zeit von Besl und McKay (1992), Chen und Medioni (1992) und Zhang (1994) entwickelt. Es ist ein Point Set Registration Algorithmus, um zwei Punktwolken auszurichten. Gegeben sind die Punktwolken des Data Set P und des Model Set Q. N_P und N_Q bezeichen die Anzahl der Punkte in der jeweiligen Punktwolke. In iterativen Schritten wird eine starre Transformation $T(R,t)$ zunächst berechnet und anschließend auf P angewand, um eine Kostenfunktion D zu minieren. Die Punktwolke P wird somit nach Q ausgerichtet. Die Kostenfunktion D ist die euklidische Distanz zwischen den beiden Punktwolken. Für jeden Punkt $p \in P$ wird der jeweilige engste Punkt in Q ermittelt. Die summierte euklidische Distanz dieser engsten Punktpaare ergibt die Distanz zwischen den Punktwolken und somit die Kostenfunktion D. Jede Iteration des Algorithmus besteht aus folgenden Schritten:

1. Berechne die engsten Punktpaare.
2. Nutze die Punktpaare, um eine Transformation T zu berechnen, welche die Kostenfunktion D verringert.
3. Wende T auf P an.
4. Beende Iteration, wenn die Änderung der Kostenfunktion D unter den vorher festgelegten Schwellwert τ fällt.

Besl und McKay (1992) haben bewiesen, dass dieser Algorithmus die Distanz zwischen den Punktwolken in jedem Schritt monoton verringert und der Algorithmus schließlich in einem Minimum der Kostenfunktion endet. Zu beachten ist jedoch, dass der Algorithmus lediglich zu einem lokalen Minimum konvergiert, welches nicht zwangsläufig das globale Minimum ist. Somit ist es möglich eine inkorrekte Ausrichtung zu berechnen.

Über die Jahre wurden verschiedene Abwandlungen und Optimierungen für den ICP Algorithmus entwickelt. Einige davon werden in den nachfolgenden Kapiteln erläutert.

6.2.2 Berechnung der engsten Punktpaare

Die ursprüngliche naive Implementation zur Suche der engsten Punktpaare vergleicht jeden Punkt $p \in P$ mit jedem Punkt $q \in Q$. Daraus resultiert eine Laufzeit von $O(N_P * N_Q)$ beziehungsweise $O(n^2)$. (Besl and McKay 1992, Nüchter 2009, 52 und 64)

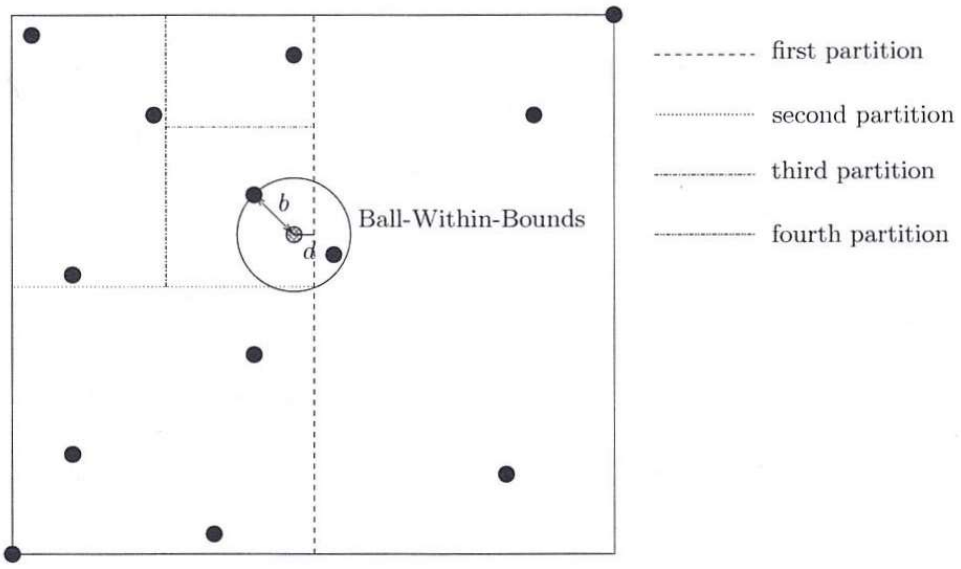


Abbildung 6-4: k-d Tree mit $k = 2$ und Ball-Within-Bounds Test (Nüchter 2009, 54)

Die Paarfindung ist die klassische k-Nearest-Neighbor Problematik mit $k = 1$. Um die Suche zu optimieren kann ein k-d Tree eingesetzt werden. In diesem Fall ist $k = 3$, womit es sich um einen drei dimensionalen Baum handelt. Dabei wird zunächst in $O(n)$ im drei dimensionalen Raum ein Suchbaum konstruiert. Für die Konstruktion wird eine Axis-Aligned Bounding Box erzeugt, welche alle Punkte beinhaltet. Anschließend wird diese Box und somit die Punktmenge solange mit Axis-Aligned Hyperplanes aufgeteilt, bis in jedem resultierenden Quader nur eine vorher definierte maximale Anzahl an Punkten vorliegt. Abbildung 6-4 zeigt das Verfahren mit $k = 2$ für einen zwei dimensionalen Raum. In dem resultierenden Bum bestehen die inneren Knoten aus den Hyperebenen, während die Blätter die Punkte beinhalten. Nach der Konstruktion kann dieser Baum für die Suche nach dem nächsten Nachbarn genutzt werden. Zunächst wird das Blatt gesucht in dem sich der Punkt befindet. Danach wird der Punkt lediglich mit den Punkten in dem Blatt verglichen und die kürzeste Distanz berechnet. In einem weiteren Schritt, wird die kürzeste Distanz

außerdem mit der Distanz des Punktes zu den umliegenden Hyperebenen verglichen. Ist die Distanz zu einer Hyperebene geringer als die zu einem Punkt in dem Blatt, muss der Punkt weiterhin mit den Punkten in dem mit der Hyperebene benachbarten Quader verglichen werden. Dies wird auch als Ball-Within-Bounds Test bezeichnet. Die Suche in einem k-d Tree beläuft sich auf eine Laufzeit von $O(\log n)$. Im Verbund mit der Konstruktion des Baumes ergibt sich eine Gesamtlaufzeit von $O(n * \log n)$. Weiterhin lässt sich der k-d Tree über die maximale Anzahl der Punkte in den Blättern und der Auswahl der Hyperebenen optimieren. Für den Einsatz im ICP Algorithmus haben Greenspan und Yurick (2003) den Approximate k-d Tree entwickelt. Bei diesem Verfahren wird lediglich innerhalb eines Blattes nach dem nächsten Nachbar gesucht und der Ball-Within-Bounds Test wird ausgelassen. In ihrer Arbeit bewiesen sie, dass der ICP trotzdem in einem lokalen Minimum endet, welches zudem sehr nahe bei dem tatsächlichen Minimum liegt. (Nüchter 2009, 52 ff.)

Eine weitere Optimierung speziell für nur partiell überlappende Daten, ist es bei der Suche nach dem nächsten Nachbarn eine maximale Distanz anzugeben. Lässt sich innerhalb dieser Distanz kein Nachbar finden, so verbleit der Punkt ohne einen und wird nicht in die Berechnung der Transformation T oder Kostenfunktion D einbezogen. Diese Optimierung spiegelt die tatsächliche Datenlage genauer wieder, da bei nur partiell überlappenden Punktwolken nicht jeder Punkt im Data Set P einen zugehörigen Punkt im Model Set Q besitzt. Laut Nüchter (2009, 69 f.) verbessert dies das Ergebnis der Ausrichtung für partiell überlappende Daten.

6.2.3 Berechnung der Transformation

Für die Berechnung der Transformation T gibt es eine Vielzahl von verschiedenen Ansätzen und Lösungen. Diese lassen sich in direkte und indirekte Methoden unterscheiden. Direkte Methoden bezeichnen dabei Lösungen die eine mathematisch geschlossene Form darstellen, wie die Singulärwertzerlegung (SVD für Singular Value Decomposition) nach Arun, Huang und Blostein (1987). Der Vorteil von direkten Methoden ist, dass sie schneller sind als indirekte Methoden (Nüchter 2009, 36 f.). Indirekte Methoden hingegen liefern bei ungleichen Punktwolken robustere und genauere Ergebnisse bei der Point Set Registration (Chen and Medioni 1992, Zhang 1994, Xu, Jiang and Chen 2012). Zu den indirekten Methoden gehören unter anderem die Modellierung von physikalischen Federn zwischen den Punktpaaren, die Abwandlung Point-To-Plane nach Chen und Medioni (1992) und Plane-To-Plane, auch Generalized-ICP, nach Segal, Haehnel und Thrun (2009). Bei den Abwandlungen Point-To-Plane und Plane-To-Plane werden für Berechnung der Distanz Oberflächen und keine Punkte genutzt. Indirekte Methoden werden häufig mit dem Levenberg-Marquardt Algorithmus gelöst (Nüchter 2009, 37).

6.2.4 Verringerung der Punkte

Die in Kapitel 6.2.2 beschriebe Laufzeit für die Berechnung der engsten Punktpaare von $O(n^2)$ oder im optimierten Fall von $O(n * \log n)$ ist die ausschlaggebende Zeitkomplexität für den ICP. Die Berechnung der Transformation T und der Anwendung von T auf P besitzen jeweils lediglich eine Komplexität von $O(n)$. Somit kann durch die Reduzierung der Anzahl der Punkte die benötigte Zeit zum Ausführen der Point Set Registration maßgeblich verringert werden (Nüchter 2009, 64). Zur Reduzierung können verschiedene Verfahren angewandt werden, wie das nachträgliche Anwenden des in Kapitel 6.1.3 erläuterten Voxelgitter Filters oder bereits bei der Aufnahme mit dem in Kapitel 6.1.5 erläuterten TSDF Voxelgitter Filter die Punktmenge reduzieren. Durch solche Filter wird zudem sowohl das Rauschen als auch die Anzahl an Ausreißern minimiert, was wiederum die Punktpaarung verbessert (Lee, Woo and Suk 2001).

6.2.5 Vorausrichtung

Um die benötigten Iterationen zu verringern kann eine Vorausrichtung vorgenommen werden. Dies verringert nicht nur die benötigte Zeit des Algorithmus, sondern ist auch ausschlaggebend dafür, dass der ICP zu dem globalen Minimum konvergiert und eine korrekte Point Set Registration durchführt (Nüchter 2009, 69). Dabei werden weitere meist Anwendungsfall abhängige externe Informationen genutzt, um eine möglichst gute anfängliche Schätzung der Transformation T zu erstellen. Zum Beispiel beschreiben Xu, Jiang und Chen (2012) in ihrer Arbeit die Problematik lokale Karten von mehreren Robotern auszurichten. Die bisherigen Verfahren zur Ausrichtung waren durch Rauschen und Messungenauigkeiten ungenau. Ihr Ansatz ist es die bisherigen Ausrichtungsmethoden als Vorausrichtung zu nutzen, um dann anschließend mit einem point-to-line ICP die Genauigkeit zu erhöhen.

6.3 Simultaneous Localization and Mapping

Simultaneous Localization and Mapping (SLAM) bezeichnet sowohl die Problematik eines Agenten, welcher in einer unbekannten Umgebung startet und dort navigieren soll, als auch das Verfahren zur Lösung der Problematik. Die Thematik entstammt ursprünglich aus der Robotik und Navigation, ist jedoch auf allgemeine Agenten aller Art anwendbar, wie auch Tiere, Menschen und Augmented Reality Geräte. Der Agent soll dabei eine Karte der Umgebung erstellen und gleichzeitig seine Position in dieser Umgebung feststellen können. SLAM ist dabei lediglich der Oberbegriff für die abstrakte Problematik und das abstrakte Verfahren zum Lösen. Konkrete Verfahren und Lösungen sind für jeden Anwendungsfall individuell zu gestalten. (Milford 2008, 9 f., Stachniss 2009, 3, Chong, et al. 2015)

Die Schwierigkeit der SLAM Problematik wird von häufig als Henne-Ei-Problem dargestellt. Um eine Karte zu erstellen, muss der Agent wissen wo er sich befindet. Um sich zu lokalisieren, benötigt der Agent jedoch eine Umgebungskarte. Wenn der Agent in einer unbekannten Umgebung startet, ist weder eine Karte noch die Position gegeben und der Agent muss beides gleichzeitig erstellen und feststellen. Die Fähigkeit in einer unbekannten Umgebung ohne bekannte Position zu starten ist aber gleichzeitig auch der große Vorteil von SLAM Systemen gegenüber herkömmlichen Systemen, die vorab Informationen über ihre Umgebung oder Position benötigen. Die SLAM Problematik gilt als eine der schwierigsten aber zugleich auch als eine der fundamentalsten Aufgaben, um sich vollständig autonom bewegende Roboter zu realisieren (Dissanayake, Durrant-Whyte and Bailey 2000, Leonard, Newman and Rikoski, et al. 2001, Leonard and Newman, Consistent, Convergent, and Constant-Time SLAM 2003, Chong, et al. 2015). (Milford 2008, 11, Stachniss 2009, 3 f.)

Ein SLAM System muss verschiedene Verfahren unterstützen, wie Odometrie, Raumwahrnehmung, Matching, Lagebestimmung und Aktualisierung der Karte. Ein häufiger Aufbau ist, über bestimmte Sensoren Odometrie zu betreiben, während weitere Sensorsystem die Umgebungskarte skizzieren. Da jedoch durch verrauschte Messungen die Odometrie langfristig ungenau wird und über diese Abweichung ein Drift entsteht, muss dieser Drift korrigiert werden. Dazu wird eine explizite Lokalisierung über die Umgebungskarte durchgeführt und mittels Kalman Filter mit der Odometrie zusammengeführt. Eine verbleibende Ungewissheit über die tatsächliche Position bleibt jedoch weiterhin bestehen. Auch die Sensoren für die Kartografierung der Umgebung sind nicht rauschfrei und die Methoden zur Einarbeitung der Messung in die Karte produzieren ebenfalls Abweichungen. Somit kann die Umgebungskarte ebenfalls einen Drift beinhalten. (Milford 2008, 9 ff., Chong, et al. 2015)

Für SLAM Systeme stellen Schleifen eine besondere Problematik dar. Ein Agent muss erkennen können, wenn dieser über eine Schleife an einen bereits besuchten Ort gelangt. Aufgrund des allgemeinen Drifts wird die Umgebungskarte des Agenten zwangsläufig wie in Abbildung 6-5 links verzerrt sein. Über die Karte kann somit nicht festgestellt werden, dass sich der Agent in einem bereits kartografierten Bereich befindet. Der Agent würde den Bereich als Neu betrachten und eine fehlerhafte Kartografierung fortführen. Um diese Problematik zu lösen, müssen zusätzliche Verfahren eingesetzt werden, um bereits besuchte Orte zu erkennen. Wurde somit eine geschlossene Schleife festgestellt, kann die Karte entsprechend angepasst und wie in Abbildung 6-5 rechts entzerrt werden. Die Entzerrung korrigiert damit die fehlerhafte Umgebungskarte. Die aktive Suche und Schließung von Schleifen wird in der Robotik als allgemeine Drift Korrektur eingesetzt. Je mehr Schleifen geschlossen und bereits bekannte Orte besucht werden, desto effektiver wird der allgemeine Drift aus der Umgebungskarte eliminiert. Da das Besuchen von bereits bekannten Orten die Effizienz von Erkundungsmissionen verringert, muss individuell die Effizienz gegenüber der Genauigkeit der Karte abgewogen werden. (Ho und Newman 2006, Stachniss 2009, 117 f.)

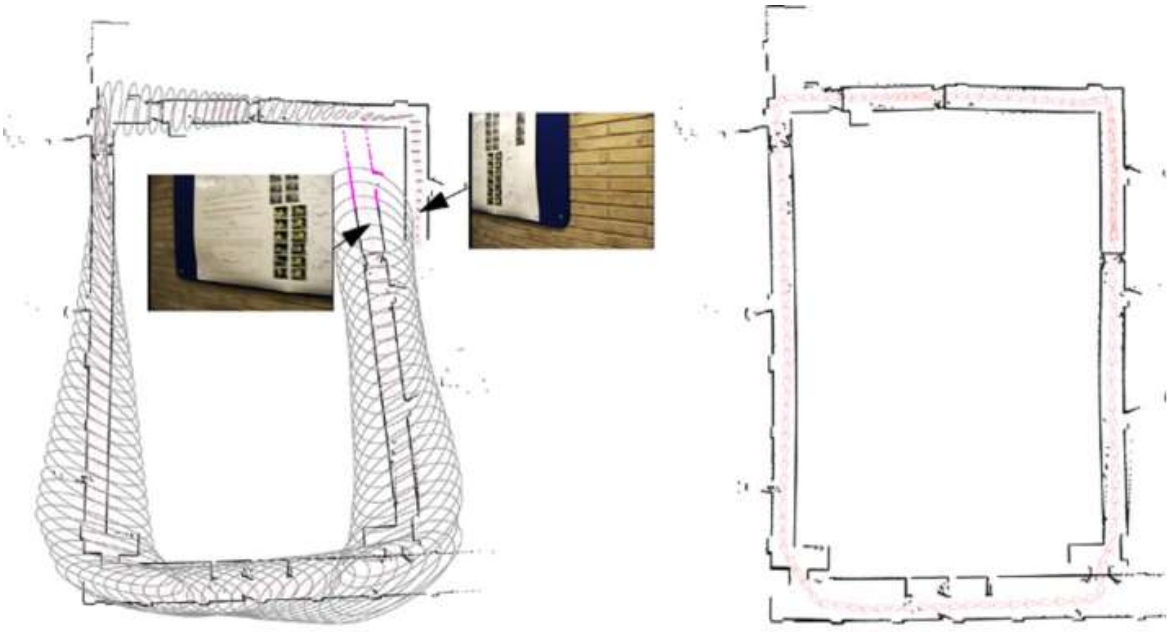


Abbildung 6-5: Drift Korrektur anhand einer Schleife (Ho and Newman 2006)

6.4 Microsoft HoloLens

Die HoloLens ist eine von Microsoft entwickelte Mixed Reality Brille. Die Brille verfügt über ein durchsichtiges Display und ist in der Lage sowohl die Umgebung als Polygonmodell zu erfassen, als auch die eigene Position und Rotation in diesem Raummodell festzustellen. Im Gegenteil zu den herkömmlichen Virtual Reality Brillen, ist die HoloLens ein eigenständiges Gerät und ist nicht auf einen weiteren Computer zur Nutzung angewiesen. Neben einem 32 Bit Intel Hauptprozessor ist, auch ein eigens von Microsoft entwickelter Coprozessor, namens Holographic Processing Unit (HPU), verbaut. Dieser soll die Sensor Daten für das Erfassen der Position und Umgebung verarbeiten und den Hauptprozessor entlasten. Als Betriebssystem wird ein Windows 10 genutzt, sodass als Anwendungen Universal Windows Apps (UWP) zum Einsatz kommen. Zur drahtlosen Kommunikation unterstütz das Gerät sowohl Bluetooth als auch Wireless LAN. Neben dem Erfassen der Position und Umgebung, besitzt das Gerät ebenfalls die Fähigkeit zur Sprach- und Gestenerkennung, räumliche Ausgabe von Sound und ist zudem mit einer Webcam ausgestattet. Weiterhin verfügt die HoloLens über vier Umgebungskameras, einem IMU und eine Tiefenkamera. (Microsoft Corporation 2017)

Das Erfassen der Position und der Umgebung ist ein SLAM System. Die vier Umgebungskameras und der IMU dienen für dieses als Sensoren. Für das Verfolgen der Position wird Visual Inertial Odometry (VIO) eingesetzt. Bei diesem wird der IMU zum kurzfristiges tracken eingesetzt. Die vier Umgebungskameras werden zum langfristigen Tracking per Viusal Odometry eingesetzt. Dabei werden Bildmerkmale in mehreren aufeinander folgenden Bildern verfolgt, um eine Aussage über die Bewegung der Kamera zu treffen. Beide Einzelsysteme werden per Kalman Filter zusammengeführt, um so die Vorteile beider Systeme zu nutzen und deren Nachteile auszumerzen. Für das Umgebungsmodell werden Tiefeninformationen zur Umgebung benötigt. Dafür werden ebenfalls der IMU und die Umgebungskameras genutzt. Zwei konsekutiv aufgenommene Bilder werden dafür als Stereokamerasystem verwendet. Über die vorherige Positionsermittlung lässt sich der Versatz beider Bilder bestimmten. Genauere Details über die eingesetzten Verfahren der HoloLens, wie die Rekonstruktion, sind öffentlich nicht bekannt. Microsoft hat jedoch in den vergangenen Jahren Publikationen zur Rekonstruktion mit der Microsoft Kinect (Newcombe, et al. 2011) oder einer Monokamera (Pradeep, et al. 2013) veröffentlicht. Somit liegt nahe, dass die HoloLens ebenfalls diese beziehungsweise darauf aufbauende Verfahren nutzt. Beide Publikationen nutzen das TSDF Verfahren um die Umgebungsdaten zu speichern und erstellen die Polygonrepräsentation per Raycasting. (Miesnieks 2017)

6.5 Google Tango

Tango ist eine Plattform für Android Geräte um Augmented Reality Anwendungen zu entwickeln. Android Geräte müssen dabei speziell Tango kompatibel sein. Dies bedeutet, dass diese über bestimmte Hardware, wie Weitwinkelkamera, Tiefenkamera und genaue Sensoren Zeitmessung verfügen müssen. Hinzu kommt, dass für das Gerät ein entsprechendes Tango Software Packet zur Verfügung stehen muss. Diese Software Packet ermöglicht es die genaue Position und Orientierung des Gerätes zu verfolgen, per sogenanntem Area Learning lässt sich die Genauigkeit erhöhen und über ein 3D Rekonstruktion Packet lässt sich die Umgebung als Polygonmodell exportieren. Die Tango Plattform stellt somit ein SLAM System dar. (Google Inc. 2017)

Für das Erfassen und Verfolgen der Position wird wie bei der HoloLens VIO eingesetzt. Das Verfahren ist nahezu identisch, mit der Ausnahme, dass die Tango Plattform lediglich eine Umgebungskamera einsetzt (Miesnieks 2017). Für die 3D Rekonstruktion der Umgebung wird die Tiefenkamera eingesetzt. Die Tiefenbilder werden dafür mit der in Kapitel 6.1.5 beschriebenen Methode per TSDF zusammengeführt. Mit einem Marching Cubes Algorithmus wird anschließend ein Polygonmodell erzeugt. (Dryanovski 2016)

7. Tracking System

Für das Tracking System müssen alle Geräte, in diesem Fall die HoloLens und der Handcontroller, eigenständige SLAM Systeme für den Innenbereich sein. Jedes Gerät erfasst eigenständig seine Position, Orientierung und ein Umgebungsmodell als lokale Karte. In einer Startphase wird nun zunächst ein globaler Referenzrahmen geschaffen. Dazu werden die lokalen Umgebungskarten der Geräte zu einer globalen Umgebungskarte ausgerichtet. Diese Ausrichtung resultiert in einer Transformation, um die Punkte und die Orientierungen von der vorher ausgerichteten lokalen Umgebungskarte in die globale Umgebungskarte zu übertragen. Über die inversen Transformationen können ebenfalls Punkte und Orientierungen von der globalen Umgebungskarte in eine lokale Umgebungskarte übertragen werden. Dies kann nach der Startphase genutzt werden, um die Position und Orientierung des Handkontrollers vom lokalen Referenzrahmen über den globalen Referenzrahmen der HoloLens zu transformieren. Da zu Beginn jedoch keine globale Umgebungskarte existiert, kann und muss dies vereinfacht werden, indem der Referenzrahmen eines Gerätes als Global deklariert wird. Für die HoloLens ist eine ständige Umgebungskarte für den normalen Gebrauch notwendig, selbst wenn keine Anwendung läuft (Microsoft Corporation 2017). Zudem verfügt die HoloLens mit ihren vier Umgebungskameras über eine sehr hohe Sensoren Abdeckung. Somit ist eine umfangreiche Umgebungskarte beim Starten der Anwendung bereits nach sehr kurzer Zeit verfügbar (s. Abbildung 7-1). Hinzu kommt, dass eine Master/Slave Relation zwischen der HoloLens und dem Handcontroller besteht. Die HoloLens ist das primäre Gerät, während der Handcontroller lediglich als Zusatzgerät fungiert. Für den Gebrauch benötigt die HoloLens die Position und Orientierung des Handcontrollers, der Handcontroller jedoch nicht die Position und Orientierung der HoloLens. Weiterhin besteht so die Möglichkeit einfach einen weiteren Handcontroller zum System hinzuzufügen und diesen unabhängig vom anderen Handcontroller an der Karte der HoloLens auszurichten. Daher wird das Referenzsystem der HoloLens als Global deklariert.

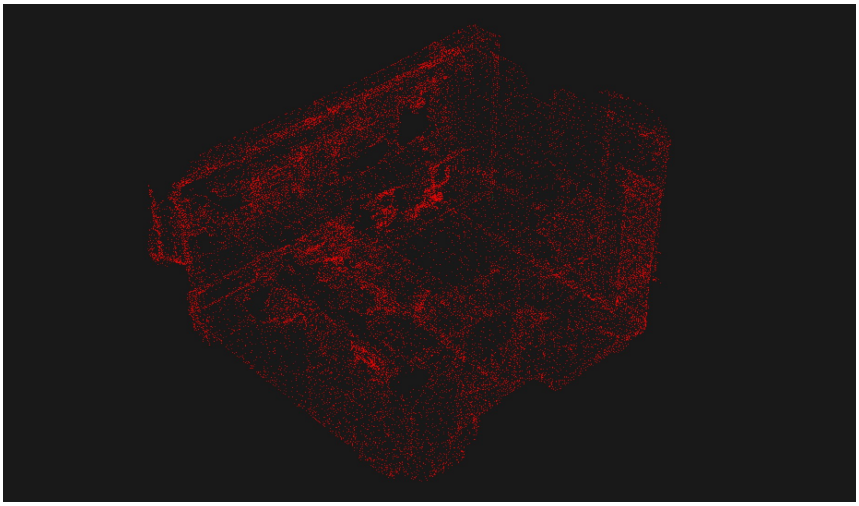


Abbildung 7-1: HoloLens Punktwolke bei Anwendungsstart

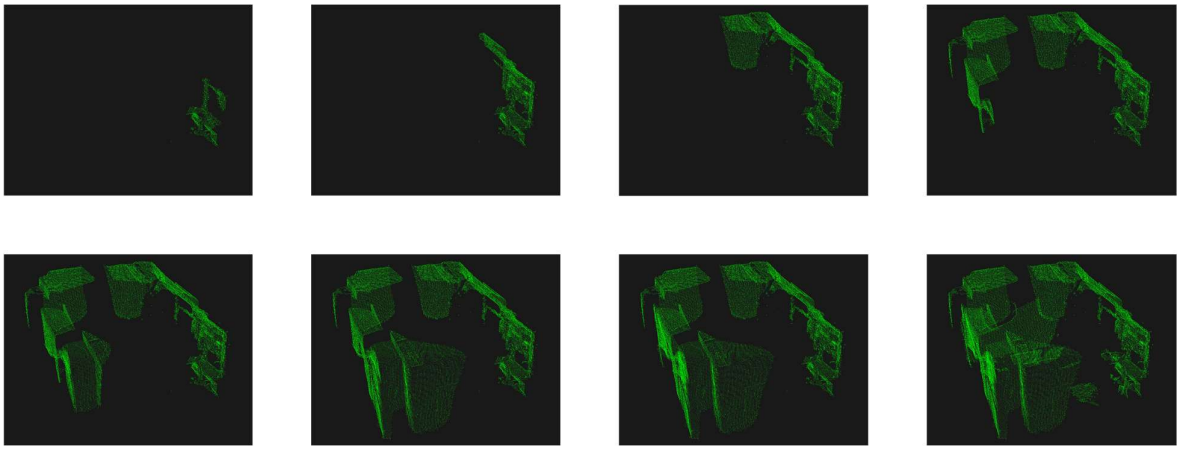


Abbildung 7-2: Konsekutiver Raumscan des Handcontrollers in Sekundenabständen

Das Ausrichten von Umgebungskarten ist eine bekannte Problematik in der Robotik. Dort wird es als Map Alignment bezeichnet und wird bei Systeme mit mehreren kooperierenden Robotern benötigt. Diese Roboter stellen ebenfalls eigenständige SLAM System dar, die eine lokale Karte anlegen. Um Roboter oder Aufgaben zu koordinieren wird eine globale Karte benötigt, bestehend aus den lokalen Karten der Roboter. Zusätzlich muss jeder Roboter seine Position in dieser globalen Karte kennen. Die Ausrichtung wird dabei in zwei Schritten ausgeführt. Zunächst wird entweder mithilfe von Landmarken oder über ein Rendezvous und direkter Sensorerkennung des anderen Roboters eine grobe Ausrichtung vorgenommen. Anschließend wird ein ICP Algorithmus eingesetzt, um die Karten endgültig und genau auszurichten. An diesem Verfahren wird sich orientiert. (Ballesta, et al. 2010, Xu, Jiang und Chen 2012)

Da der Handcontroller zwangsläufig klein und in der Hand haltbar sein muss, kann davon ausgegangen werden, dass dieser nur eine sehr kleine Sensorabdeckung erreicht. Somit wird die Umgebungskarte des Handcontrollers mittels eines Raumscans konsekutiv wie in Abbildung 7-2 erstellt und ist nicht wie bei der HoloLens zeitnah vollständig verfügbar. Dadurch gestaltet sich eine Vorausrichtung per Landmarken als schwierig. Auch eine Vorausrichtung per Rendezvous setzt sowohl einen direkten Sichtkontakt, als auch eine fortgeschrittene Objekterkennung voraus. Dennoch soll eine Vorausrichtung getätigt werden, um zum einen den Ausrichtungsprozess zu beschleunigen, und zum anderen eine Falschausrichtung zu vermeiden. Denn Innenräume sind in der Regel rechteckig und um 180 Grad gedrehte oder je nach Seitenverhältnis können auch um 90 Grad gedrehte Raumkarten lokale Minima der Kostenfunktion des ICP darstellen. Stattdessen wird die Gravitation und die Kompassrichtung zur Vorausrichtung genutzt. Der IMU der HoloLens besteht aus Beschleunigungssensor, Magnetometer und Gyroskop (Holmdahl 2015). Über den Beschleunigungssensor lässt sich die Richtung der Gravitation und über das Magnetometer die Richtung des geomagnetischen Nordpols bestimmen. Mit zwei nicht kollinearen Vektoren lässt sich eine eindeutige Orientierung bestimmen. Bei dem Referenzrahmen der HoloLens wird bereits die y-Achse als entgegengerichtet zur Gravitation definiert. Um die Ausrichtung zu vereinfachen, soll auch der Handcontroller seine Umgebungskarte so anlegen, dass die y-Achse entgegengerichtet der Gravitation ist. Dadurch muss die Vorausrichtung lediglich die Kompassrichtung beachten.

Der Ablauf für das Ausrichten wird auf Abbildung 7-3 gezeigt. Der Prozess des Ausrichtens wird mit Raum Ausrichtung angegeben. Zu Beginn startet die Raum Ausrichtung mit der HoloLens. Die HoloLens übergibt der Raum Ausrichtung die Punktwolke ihrer Umgebung. Da die HoloLens die Umgebungskarte kontinuierlich aktualisiert, werden diese ebenfalls übertragen, damit die Raum Ausrichtung jederzeit eine aktuelle Kopie der Punktwolke besitzt. Zu einem späteren Zeitpunkt wird der Handcontroller eingeschaltet. Dieser sendet ein Startsignal, welchem die Nordrichtung beigefügt ist. Die Nordrichtung ist dabei der Winkel θ zwischen magnetischem Norden und dem Vorausvektor des lokalen Referenzrahmens in der xz-Ebene. Die HoloLens empfängt das Start Signal und sendet ebenfalls die Nordrichtung in ihrem lokalen beziehungsweisen globalen Referenzrahmen. Die Raum Ausrichtung erhält die beiden Werte $\theta_{Handcontroller}$ und $\theta_{HoloLens}$. Mit

$$\theta_{Vorausricht} = \theta_{Handcontroller} - \theta_{HoloLens}$$

lässt sich der Winkel zwischen den beiden lokalen Referenzsystemen berechnen. Über die Rotationsmatrix zur Rotation um die y-Achse

$$R_y(\theta) = \begin{matrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{matrix}$$

lässt sich so die Transformation $T(R,t)$ zur Vorausrichtung berechnen. Eine Translation t wird bei der Vorausrichtung nicht berechnet. Nach dem Senden des Start Signals beginnt der Handcontroller ebenfalls die Punktwolke seines Raumscans zu übermitteln. Da der Handcontroller einen fortlaufenden Raumscan durchführt, müssen ständige Aktualisierungen und Ergänzungen übermittelt werden. Daraufhin startet die Raum Ausrichtung das Point Matching per ICP. Dabei gilt die Raumpunktwolke der HoloLens als Model Set und die des Handcontrollers als Data Set. Nachdem der ICP die Konvergenz erreicht hat, wird die Transformation $T(R,t)$ an die HoloLens übermittelt. Da der Handcontroller einen fortlaufenden Raumscan anfertigt, soll der ICP abgewandelt werden, um die Aktualisierungen möglichst zeitnah zu berücksichtigen. Dazu wird der ICP um den folgenden Schritt erweitert. Zu Beginn der Iteration, bevor die Punktpaare ermittelt werden, werden das Data Set und das Model Set entsprechend den Aktualisierungen angepasst. Somit arbeitet jede Iteration mit aktuellen Kopien der Umgebungskarten. Anderenfalls könnte zunächst eine fehlerhafte Ausrichtung berechnet werden, da vom Handcontroller zu Beginn unzureichende Daten vorliegen. Dieser Fall könnte auch eintreten, wenn der ICP schneller konvergiert, als Umgebungsaktualisierungen vom Handcontroller übermittelt werden. Um dies zu verhindern, wird der ICP nach jeder Konvergenz sobald neue Aktualisierungen vorliegen erneut gestartet. Somit wird der Ausrichtungsprozess solange durchlaufen, bis der Raum ausreichend vom Handcontroller gescannt wurde, dass trotz Aktualisierungen keine genauere Konvergenz erreicht werden kann. Da die Raum Ausrichtung nicht feststellen kann, ob der Raum vollständig gescannt wurde oder noch Aktualisierungen zu erwarten sind und ob nun eine korrekte und ausreichende Ausrichtung erreicht wurde, muss der Nutzer visuell eine ausreichende Ausrichtung feststellen und die Startphase manuell beenden.

Voraussetzung für dieses Verfahren ist, dass beide SLAM Systeme den Raum ausreichend und überlappend scannen. Da der Handcontroller den Raum nicht sofort und vollständig erfasst, müssen das Data und Model Set zunächst als überlappend betrachtet werden. Daher muss eine maximale Distanz zum Herstellen der Punktpaare verwendet werden. Weiterhin sind die Punktwolken nicht fix, sondern werden während des Algorithmus verändert. Die hinzukommenden Punkte können aber in den vorherigen Iterationen als außerhalb der maximalen Distanz zu einem Partner betrachtet werden. Daher kann davon ausgegangen werden, dass der ICP weiterhin in ein lokales Minimum konvergiert.

Nachdem eine Transformation $T(R,t)$ von der Raum Ausrichtung berechnet wurde, beginnt der Handcontroller fortlaufend seine Position und Orientierung an die HoloLens zu übermitteln. Die Transformation wird ebenfalls an die HoloLens übergeben und kann daraufhin genutzt werden um die Position und Orientierung des Handcontrollers in den globalen beziehungsweise lokalen Referenzrahmen der HoloLens zu übertragen. Über die Inverse Transformation $T^{-1}(R,t)$ könnte auch der Handcontroller die Position und Orientierung der HoloLens in seinen lokalen Referenzrahmen übertragen.

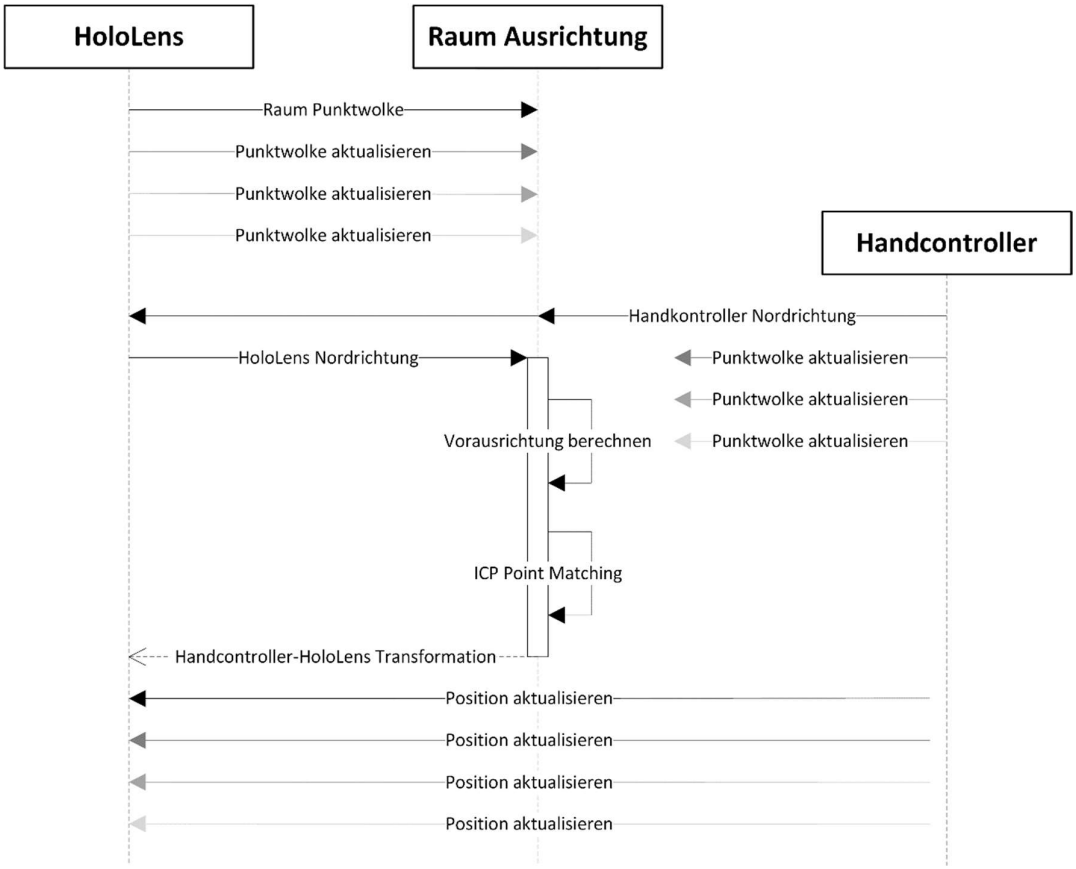


Abbildung 7-3: Ablaufdiagramm für die Raum Ausrichtung

8. Handcontroller Hardware

Zur Auswahl der Handcontroller Hardware standen entweder eine prototypische Umsetzung mit einer gängigen Tiefenkamera, wie Microsoft Kinect oder Asus Xtion Pro Live oder die Verwendung eines Lenovo Phab 2 Pro Handys (s. Abbildung 8-1). Chong, et al. (2015) und Li, Wie und Lan (2016) zeigten bereits, dass mit diesen günstigen Tiefenkameras stabile und zuverlässige SLAM System umsetzbar sind. Qayyum, Ahsan and Mahmood (2017) entwickelten ein verbessertes System, welches einen IMU einsetzt und sehr akkurate Ergebnisse für Innenbereiche liefert. Das Lenovo Phab 2 Pro Handy ist ein Google Tango kompatibles Android Gerät. Es ist ein Endverbraucher Gerät und besitzt über die Google Tango Plattform alle nötigen Funktionen für den Handcontroller. Außerdem verfügt es für die drahtlose Kommunikation über WLAN und Bluetooth (Lenovo 2017). Da es ein Android Handy ist, muss für das Gerät eine Anwendung geschrieben werden, welche wiederum weitere Berechnungen auf dem Handcontroller ermöglichen. Weiterhin besitzt es einen Touchscreen mit dem sowohl die einfache Tap Geste als auch komplexere Eingaben erfasst werden können. Zudem kann der Bildschirm auch dazu genutzt werden dem Nutzer Anweisungen anzuzeigen. Da es wie bereits beschrieben ein Endverbraucher Gerät ist, verfügt es ebenfalls über eine gewisse Ergonomie.

Nach Abwägung wurde sich dazu Entschieden das Lenovo Phab 2 Pro als Hardware für den Handcontroller zu nutzten. Es bietet bereits eine umfangreiche und stabile Implementation eines SLAM Systems an und verfügt über weitere nützliche Zusatzfunktionen, was die Entwicklung maßgeblich beschleunigt.

Abbildung 8-1: Lenovo Phab 2 Pro (Lenovo 2017)

9. Umsetzung

Für die Umsetzung der Raum Ausrichtung wird die Point Cloud Library (PCL) in Version 1.8.1 genutzt. Die PCL ist eine umfangreiche Library für Punktwolken und liefert unter anderem eine Vielzahl an Implementationen von Algorithmen für das Point Matching, Downsampling und die Rekonstruktion. Da die PCL nicht mit UWP kompatibel ist, muss die Raum Ausrichtung als eigenständige Anwendung für einen weiteren Computer umgesetzt werden. Diese ist für die Win64 Plattform umgesetzt, sodass die Raum Ausrichtung auf einem Microsoft Windows Computer ausgeführt wird. Die Software für die HoloLens ist als Unity 5.6.3 Anwendung umgesetzt. Die Unity Engine wird als Entwicklung Plattform für HoloLens Anwendung empfohlen und ermöglicht es visuelle Funktionen einfach umzusetzen (Microsoft Corporation 2017). Als Handcontroller Software muss eine Android Anwendung verwendet werden. Auf dem verwendeten Lenovo Phab 2 Pro ist Android 6.0.1 installiert. Die Anwendung wird mit dem Android SDK 26 umgesetzt. Für die Funktionen der Google Tango Plattform wird das Tango SDK 1.55 verwendet. Für die Kommunikation wurde WLAN ausgewählt. Da das System nun aus drei eigenständigen Komponenten besteht, wird auf eine direkte Kommunikation zwischen HoloLens und Handcontroller verzichtet. Stattdessen wird zur Kommunikation das Protokoll MQTT in Version 3.1 genutzt. Bei diesem wird die Kommunikation über einen zentralen Server, Broker genannt, geleitet. Diese Besonderheit wird von den in Kapitel 10 durchgeführten Tests genutzt. Als Broker wird ein Raspberry Pi 3 Model B verwendet. Der Aufbau des Systems ist wie folgt. Die HoloLens und der Handcontroller sind mit einem WLAN Access Point (WLAN AP) verbunden. Der Raspberry Pi ist ebenfalls per Ethernet Kabel mit diesem WLAN AP verbunden.

Bei der Umsetzung der Anwendung für die HoloLens wird festgestellt, dass die HoloLens das Magnetometer nicht als Kompass Sensor einer Anwendung bereitstellt und somit kein Auslesen möglich ist. Da das ursprüngliche Konzept nicht genau umgesetzt werden kann, wurden drei Ansätze als Alternativen entwickelt. Der erste Ansatz ist das Ausrichten per Sichtkontakt zwischen HoloLens und Handcontroller. Bei diesem Ansatz wird das Ausrichten per Umgebungskarte ausgelassen. Zur Ermittlung der Ausrichtungstransformation wird auf dem Bildschirm des Handys ein Bildmarker angezeigt. Die HoloLens nutzt die Webcam und die Vuforia Plattform, um den Bildmarker zu erkennen und die Position im HoloLens Referenzrahmen festzustellen. Ein weiterer Ansatz ist die Annahme, dass der Nutzer beim Starten der Handcontroller Anwendung das Display des Handys betrachtet. Somit kann angenommen werden, dass HoloLens und Handcontroller beim Starten dieselbe Ausrichtung besitzen. Dabei wird die Position von Handcontroller und HoloLens gleichgesetzt. Der letzte Ansatz ist, auf die Vorausrichtung zu verzichten. Um eine Falschausrichtung zu verhindern, soll beim Ausrichten die Karte des Handcontrollers zunächst testweise um 90, 180 und 270 Grad gedreht werden, um anschließend zu überprüft, ob eine bessere Konvergenz vorliegt. Lediglich die ersten beiden Ansätze wurden umgesetzt. Der erste Ansatz soll genutzt werden, um das Tracking Verfahren nach dem Ausrichten zu testen. Der zweite Ansatz soll dazu genutzt werden die Anforderungen an die Raum Ausrichtung zu testen.

10. Evaluation

10.1 Testumgebung

Alle Tests wurden in dem auf Abbildung 10-1 abgebildeten Büroraum durchgeführt. Der Raum ist ca. 5,00 mal 5,90 Meter groß und kann als spärlich eingerichtet bezeichnet werden. Neben den vier Bürotischen, befindet sich ebenfalls ein Wandschrank mit einem Drucker im Raum. Der Grundriss ist nicht komplett rechteckig, da eine der Ecken nach innen gerichtet ist. Weiterhin besteht eine Wand aus einer Fensterzeile mit vorgezogenem Kabelkanal als Fensterbrett.

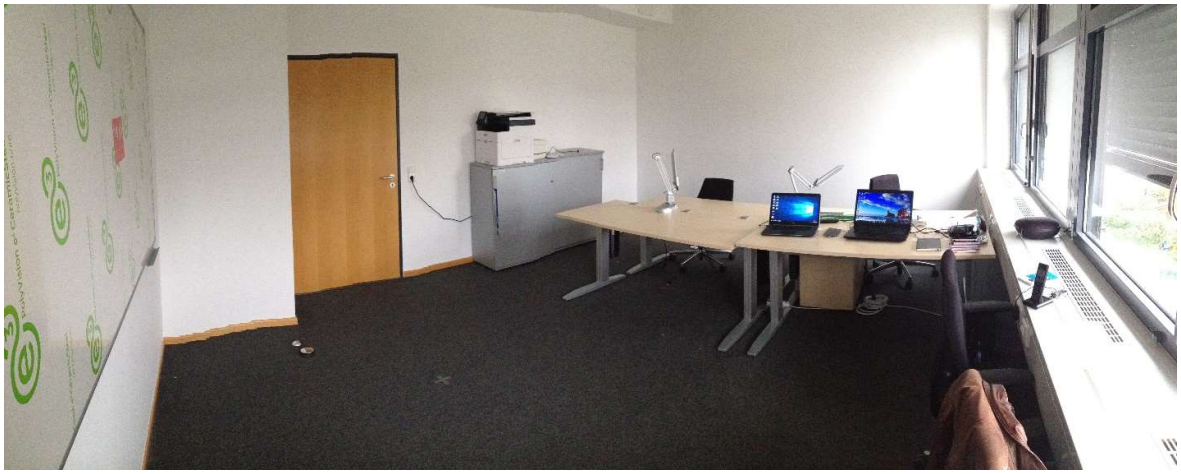


Abbildung 10-1: Testumgebung

10.2 Handcontroller

10.2.1 Verzögerung

Um die Verzögerung der Handcontroller Bewegungen zu ermitteln wird die Webcam der HoloLens eingesetzt. Zunächst werden per Sichtkontakt und Bildmarker die Referenzrahmen der HoloLens und des Handcontrollers ausgerichtet. Die Anwendung zeigt eine virtuelle Repräsentation des Handcontrollers an dessen Position an. Anschließend wird der Handcontroller in einer schnellen Bewegung zur Seite bewegt (s. Abbildung 10-2). Der Vorgang wird per Mixed Reality Capture als Video aufgenommen. Eine Mixed Reality Capture Aufnahme nimmt per Webcam ein Video auf und überblendet dies mit dem virtuellen Bild, sodass ein Video aus Sicht des Nutzers entsteht (Microsoft Corporation 2017). Mithilfe des Videos kann die Zeit zwischen Beginn der Handbewegung und Reaktion der virtuellen Repräsentation gemessen werden. Die Videos werden entgegen der Dokumentation mit 24 Bildern pro Sekunde aufgenommen, sodass die Zeit zwischen zwei Bildern ca. 42 Millisekunden entspricht. Bei der Testaufnahme werden fünf Bewegungen durchgeführt. Bei allen Testbewegungen wird eine Reaktion nach 4 Bildern festgestellt, was einer Verzögerung von ca. 167 Millisekunden entspricht.

Abbildung 10-2: Einzelbilder einer Testbewegung bis Reaktion eintritt

Derzeit erfolgt die Kommunikation über eine indirekte Verbindung. Die Daten vom Handcontroller werden zunächst über den WLAN AP zum Broker, wieder zum WLAN AP und schließlich zur HoloLens gesendet. Dies könnte man mit einer direkten drahtlosen Verbindung zwischen Handcontroller und HoloLens vereinfachen und so die Verzögerung verringern.

10.2.2 Bewegungsgeschwindigkeit

Zur Ermittlung der maximalen Bewegungsgeschwindigkeit des Handcontrollers wird ebenfalls die Ausrichtung per Sichtkontakt und Bildmarker genutzt. Das Handy wird dazu in eine Ausgangsposition an der Tischkante gebracht (s. Abbildung 10-3 links). Der Handcontroller wird nun in einem bestimmten Zeitintervall um 50 Centimeter zur Seite und wieder zurück in die Ausgangsposition bewegt. Nach jedem Test wird die Ausrichtung erneut ausgeführt. Bei Zeitintervallen von bis zu zwei Sekunden, was einer durchschnittlichen Geschwindigkeit von einem halben Meter pro Sekunde entspricht, ist keine Abweichung festzustellen. Bei Intervallen von einer bis zwei Sekunden ist anschließend eine kleine Abweichung festzustellen. Dabei scheint es, dass vor allem abrupte Beschleunigungen und Abbremsungen die Abweichungen beeinflussen. Bei Zeitintervallen von unter einer Sekunde entsteht ein sehr großer Versatz und das Tracking setzt teilweise aus. Auf eine ähnliche Weise wird auch die Rotationgeschwindigkeit getestet. Dabei wird der Handcontroller statt zur Seite bewegt um 90 Grad gedreht. Rotationen des Handcontrollers sind sehr anfällig gegenüber Drift. Selbst bei langsameren Rotationen, zwei 90 Grad Rotationen

Abbildung 10-3: Abweichung nach zwei langsamen 90 Grad Drehungen

innerhalb von zwei Sekunden, entsteht ein leichter Versatz. Erhöht man die Rotationsgeschwindigkeit nimmt entsprechend der Versatz zu. Dabei ist besonders zu beobachten, dass die Orientierung subjektiv keine Abweichung aufweist. Stattdessen ist nach Abschluss der Rotation eine kurze Versatzbewegung zur Seite wie in Abbildung 10-3 erkennbar.

Da die Google Tango Plattform VIO zum Tracking nutzt, kann vermutet werden, dass bei zu hohen Geschwindigkeiten die Visual Odometry versagt und bei zu hohen Beschleunigungen das Tracking per IMU zu ungenau wird. Weiterhin wird trotz Area Learning Modus keine ausreichende Korrektur über die Lokalisierung in der Umgebung vorgenommen.

10.2.3 Unterschiedliche Maßstäbe

Beim allgemeinen Nutzen und Testen ist ein Unterschied zwischen den Maßstäben von HoloLens und Handcontroller festzustellen. Um die genaue Abweichung zu bestimmen wird erneut eine Ausrichtung per Sichtkontakt und Bildmarker vorgenommen. Der Handcontroller wird zu Beginn in die Ausgangsposition in Abbildung 10-4 links gebracht und eine Ausrichtung wird vorgenommen. Anschließend wird der Handcontroller entlang der Tischkante um bis zu einem Meter langsam verschoben. Dabei lässt sich eine stetige lineare Abweichung abhängig von der Entfernung zum Ausgangspunkt beobachten. Nach einem Meter beträgt die Abweichung ca. 2,5 Centimeter (s. Abbildung 10-4 rechts). Es lässt sich daraus schließen, dass HoloLens und das Lenovo Phab 2 Pro unterschiedliche Maßstäbe für die zurückgelegte Strecke nutzen. Da jeweils nur ein Exemplar jeden Gerätes vorhanden ist, ist unklar, ob dies eine Plattform- oder Gerätespezifische Abweichung ist.

Dieser Unterschied zwischen den Maßstäben zeigt auf, dass eine reine Ausrichtung über Sichtkontakt und Bildmarker nicht ausreichend ist. Auch die Ausrichtung per Umgebungskarten liefert derzeit lediglich eine starre Transformation und berücksichtigt keine Maßstabsunterschiede. Eine mögliche Lösung ist einen anderen Point Matching Algorithmus für elastische Transformationen zu nutzen. Fraglich ist jedoch, ob diese die Zeitanforderung erfüllen können. Da es sich scheinbar um ein einfaches Skalierungsproblem handelt, wäre ein weiterer Ansatz den Raum Ausrichtungsprozess um ein Schritt zu erweitern. In diesem abschließenden Schritt könnte der Faktor der Skalierung berechnet werden, um die Transformation $T(R,t)$ um diesen zu $T(R,t,S)$ zu erweitern.

Abbildung 10-4: Abweichung durch Raumgrößen

10.2.4 Robustheit

Bei der allgemeinen Nutzung des Handcontrollers werden folgende subjektive Beobachtungen bezüglich der Robustheit gemacht. Wird das Handy zu nah an eine Wand herangeführt, ca. 50 Centimeter, oder in einem Abstand von 1,0 bis 1,5 Metern auf eine weiße Ecke ausgerichtet, ist ein Aussetzen des Tracking festzustellen. Anschließend ist es nicht in der Lage sich zu relokalisieren. Weiterhin treten bei Scans hin und wieder kleinere Versätze von bis zu 50 Centimetern auf. Dies äußert sich durch doppelte Wände im finalen Scan (s. Abbildung 10-5). Es ist unklar, ob diese durch zu schnelle Bewegungen oder durch kurzfristiges Aussetzen und Wiederherstellen des Trackings ausgelöst wird.

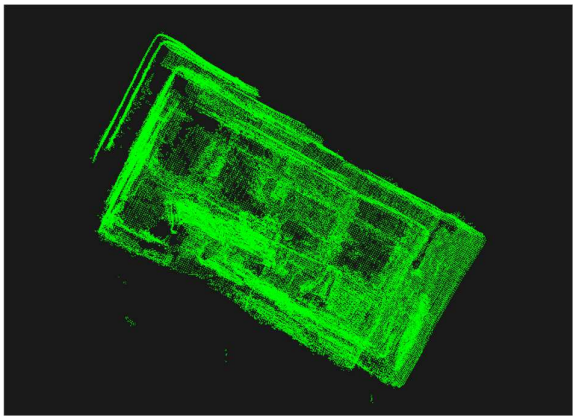


Abbildung 10-5: Fehlerhafter Scan mit doppelter Wand

10.3 Genauigkeit der Raum Ausrichtung

10.3.1 Testbeschreibung

Die PCL liefert vier verschiedene ICP Implementationen, zum einen den ursprünglichen ICP Algorithmus nach Besl und McKay (1992) mit SVD (ICP), den ursprünglichen ICP nach Besl und McKay (1992) mit Levenberg-Marquardt Algorithmus (ICP-NL), einen ICP nach Chen und Medioni (1992), welcher die Distanz von Punkt zu Ebene betrachtet, (ICP-Plane) und den Generalized-ICP (GICP) nach Segal, Haehnel und Thrun (2009). Im Test soll nun die Genauigkeit dieser vier Algorithmen verglichen werden.

Dazu wird jeweils eine Raum Ausrichtung über die Umgebungskarten mit der Annahme als Vorausrichtung vorgenommen. Danach wird der Handcontroller in einer Position fixiert, um die Abweichung zwischen realem Handy und virtueller Repräsentation zumessen. Um die Abweichung der Rotation zu erfassen, wird das Handy anschließend um 2,5 Meter entlang einer Achse verschoben und die Abweichung wird erneut gemessen. Über die Differenz der Abweichungen und der Distanz von 2,5 Metern lässt sich ein abweichender Winkel berechnen. Um das Problem der unterschiedlichen Maßstäbe zu minimieren, wird der erste Messpunkt in die Raummitte verlegt. Für die Rotationsabweichung wird lediglich der Versatz in der Ebene orthogonal zur

Verschiebungsachse betrachtet. Weiterhin werden HoloLens und Handcontroller möglichst nah beieinander und in die gleiche Richtung ausgerichtet gestartet, um weitere Abweichungen zu minimieren. Der Raumscan besteht aus drei langsamen Rotationen aus dem Stand in Brusthöhe. Zunächst nach vorne gerichtet und danach um ca. 30-45 Grad nach oben und nach unten gerichtet. Anschließend wird der Handcontroller in die erste Messposition gebracht und verbleibt dort bis der jeweilige ICP eine Konvergenz erreicht. Danach wird die Ausrichtungsphase beendet und die Messungen beginnen. Die Verschiebung zur zweiten Messposition wird als stetige langsame und konstante Bewegung durchgeführt, um den Versatz durch die Bewegungen zu minimieren.

Bei dem Test wird die maximale Distanz zwischen zwei Punkten für die Punktpaarung auf den Wert von einem Meter festgelegt. Weiterhin wird $\tau = 10^{-10}$ ausgewählt, um eine möglichst genaue Ausrichtung zu erhalten. Bei diesem Test wird kein VoxelGrid Filter eingesetzt.

10.3.2 Ergebnis

Das Ergebnis des Tests ist in Tabelle 10-1 zu sehen. Bei der Translation betragen die Abweichungen mehrere Centimeter und bei der Rotation einige Grad. Die Anforderung an die Genauigkeit der Translation beträgt 0,5 Centimeter, was von allen Algorithmen um das 22 bis 36-fache übertroffen wird. Bei der Rotation wird die Anforderung von maximal 0,11 Grad um das 17 bis 26-fache übertroffen. Somit erreicht keiner der Algorithmen die geforderte Genauigkeit. Es ist jedoch zu beachten, dass dies lediglich die Genauigkeit der Ausrichtung ist. Bewegungen werden dennoch, sofern man den Maßstabsunterschied vernachlässigt, eins zu eins abgebildet. Die empfohlene Mindestdistanz zwischen Nutzer und eingeblendetem Hologramm beträgt für die HoloLens 85 Centimeter, was größer als die Armlänge eines Menschen ist (Microsoft Corporation 2017). Daher wird ein Nutzer unter normalen Umständen nicht die virtuelle Repräsentation des Handcontrollers und somit dessen Versatz sehen können, sondern lediglich die Verlängerung des Zeigers. Es wäre zu erproben, ob die Augen-Hand-Koordination eines Menschen ausreicht, um trotz dieser Abweichung den Handcontroller erfolgreich zu nutzen.

Tabelle 10-1: Abweichungen der verschiedenen Algorithmen

ICP Algorithmus	**Translationsabweichung in Centimeter**	**Rotationsabweichung in Grad**
ICP	11,02	2,12
ICP-NL	11,07	2,86
ICP-Plane	15,11	1,90
GICP	18,00	2,20

10.4 Zeitanforderung und Raumabdeckung

10.4.1 Testbeschreibung

In einem weiteren Test soll die Anforderung der maximalen Länge der Startphase überprüft werden. Weiterhin soll ermittelt werden, wie viel Raumabdeckung beim Scannen mit dem Handcontroller benötigt wird, um eine zufriedenstellende Ausrichtung zu erhalten. Im Rahmen des Tests soll weiterhin erprobt werden, in wie weit ein VoxelGrid Filter die Raum Ausrichtung beschleunigen kann, um die Zeitanforderung einzuhalten. Dabei wird ebenfalls beachtet, dass ein zu grober VoxelGrid Filter die Genauigkeit der Ausrichtung negativ beeinflussen könnte.

Für den Test werden zwei umfangreiche Raumscans aufgenommen. Dazu werden die HoloLens und die Handcontroller Anwendungen jeweils wie in dem Verfahren vorgestellt nach einander gestartet. Mithilfe des zur Kommunikation verwendeten Protokoll MQTT lässt sich nun die Kommunikation, das heißt die Umgebungskarten Aktualisierungen, das Start des Handcontrollers und die Antwort der HoloLens, aufzeichnen. Die Aufzeichnung wird mit einer Auflösung von einer Millisekunde getätigt. Anschließend lässt sich eine Aufnahme wieder abspielen, um den durchgeführten Scan zu simulieren. Dies ermöglicht es einen einzelnen Scan mit verschiedenen Algorithmen und Parametern unter gleichen Bedingungen zu testen.

Die beiden Aufzeichnungen werden so getätigt, um einen realen Anwendungsfall nachzubilden. Auf die maximale Bewegungsgeschwindigkeit wird lediglich subjektiv geachtet. Dennoch wird sichergestellt, dass keine doppelten Wände in dem Scan auftreten. Die Aufnahmen werden aus der stehenden Position mit einer ungleichmäßigen Bewegung durchgeführt. Der Nutzer bewegt sich dabei zudem durch den Raum. Der Handcontroller wird dabei auf Brusthöhe gehalten, aber dennoch frei bewegt. Bei der Aufnahme wird primär nach vorne und schräg unten gescannt, um Tische, Gegenstände und Wände zu erfassen. Die ersten 20 Sekunden der beiden Handcontrolleraufnahmen sind in Abbildung 10-6, respektive Abbildung 10-7 zu sehen.

Zur Ermittlung der Raumabdeckung wird unabhängig von den beiden Aufnahmen ein vollständiger Scan mit HoloLens und Handcontroller durchgeführt. Dabei werden alle aus Brusthöhe sichtbaren Oberflächen erfasst. Die Punktanzahl dieser Scans gilt nachfolgend als maximale Punktzahl des Raumes für die HoloLens und den Handcontroller.

Um festzustellen, ob im entsprechenden Testdurchlauf eine ausreichende Ausrichtung erreicht wird, werden Referenztransformationen gebildet. Dies bedeutet, dass für jede Aufnahme eine Ausrichtung mit jedem verwendeten ICP Algorithmus ohne Zeitlimit und ohne die Verwendung eines VoxelGrid Filters durchgeführt wird. Nach vollständigem Abspielen der Aufnahme und einer darauffolgenden Konvergenz wird die resultierende Transformation als Referenz für eine optimale Lösung genutzt.

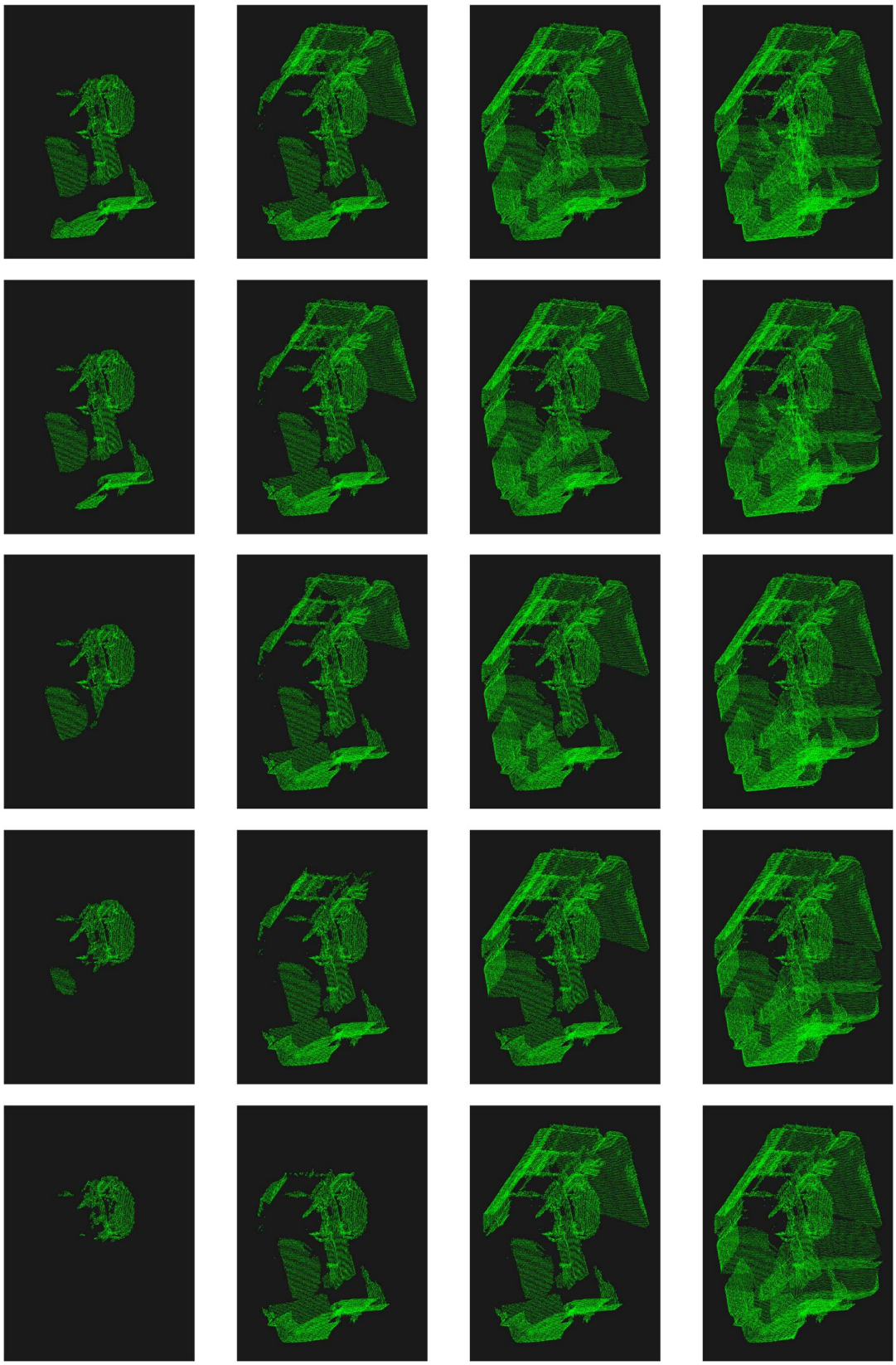


Abbildung 10-6: Handcontroller Scan der Aufnahme A

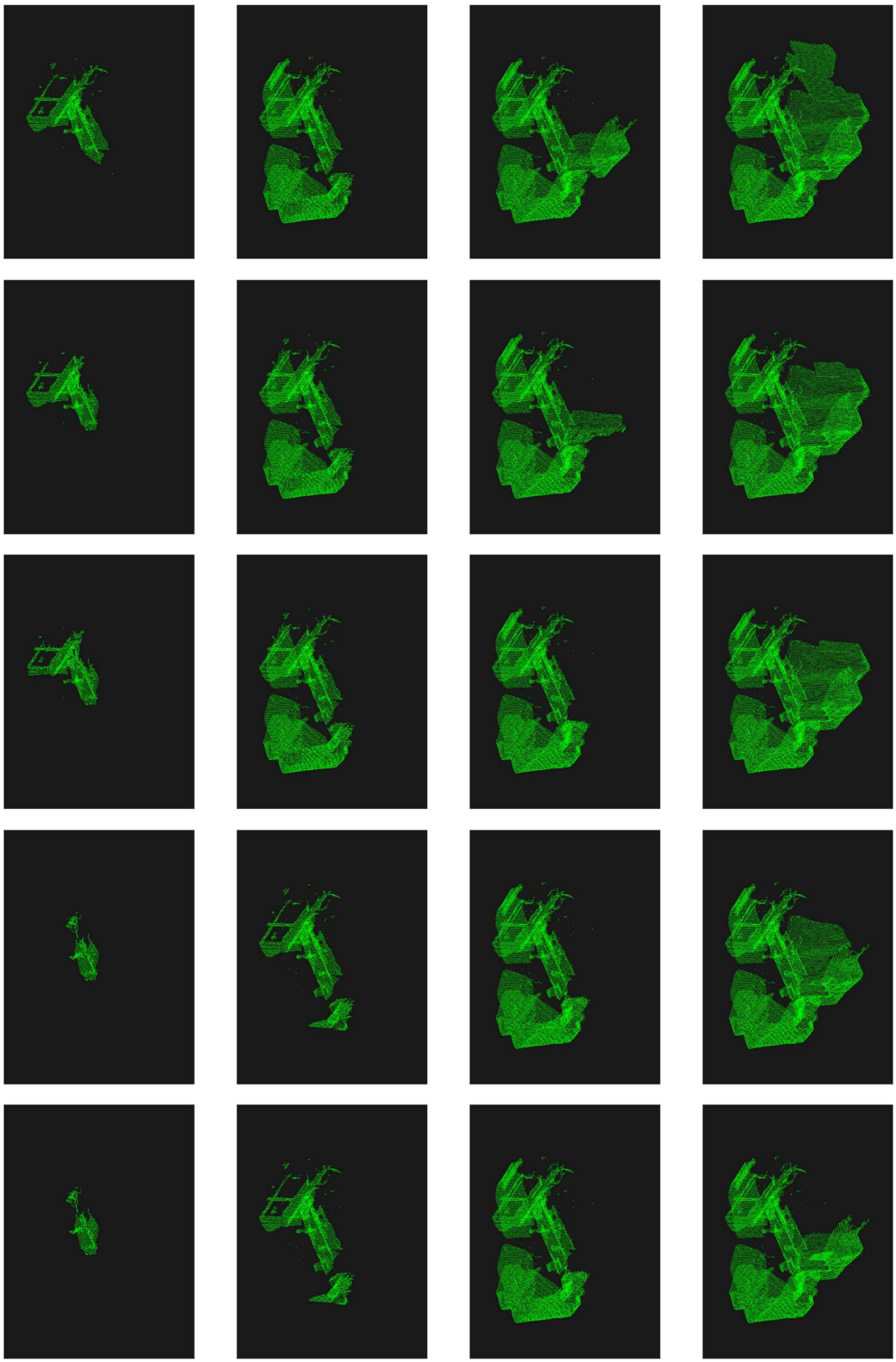


Abbildung 10-7: Handcontroller Scan der Aufnahme B

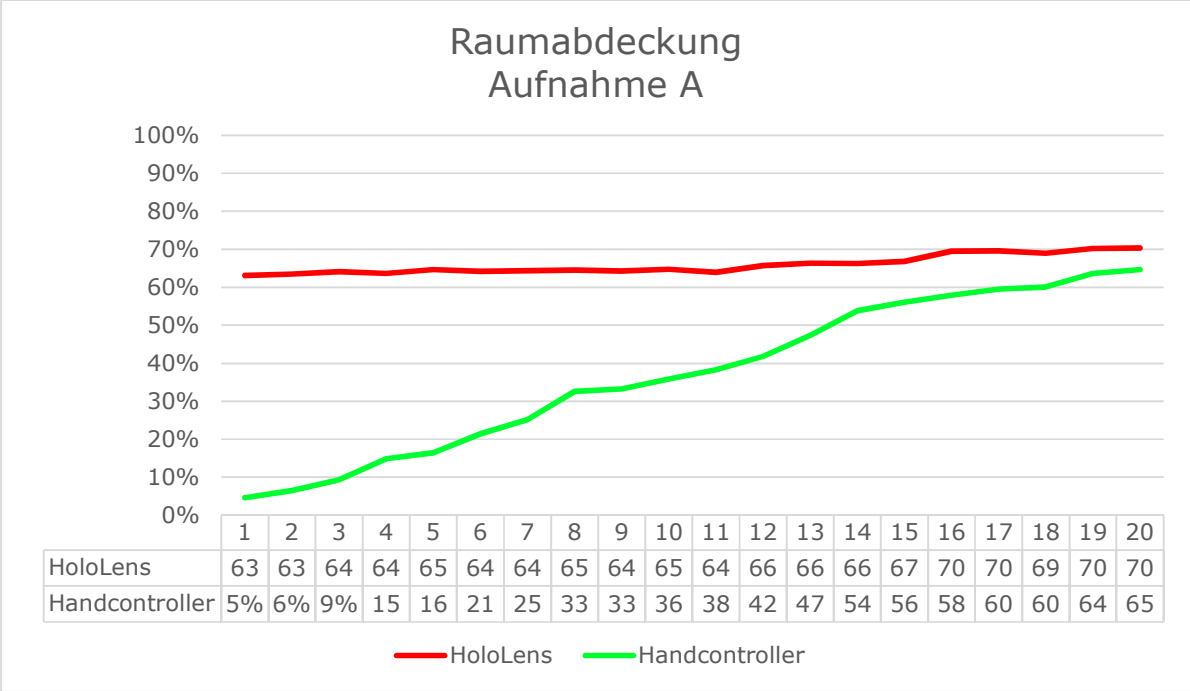
Bei dem Test wird eine Aufnahme zunächst bis eine Sekunde nach dem Start Signal des Handcontrollers abgespielt. Anschließend wird abgewartet bis die Raum Ausrichtung eine finale Konvergenz erreicht. Danach wird die Raum Ausrichtung zurückgesetzt und die Aufnahme wird nun bis zwei Sekunden nach dem Start Signal abgespielt. Dieser Vorgang wird bis zu einer Länge von 20 Sekunden nach dem Start Signal wiederholt, sodass 20 Testdurchläufe entstehen.

Nach jedem Testdurchlauf werden die benötigten Iterationen, die Zeit von Erreichen des Start Signals bis zur finalen Transformation, die Punktanzahl des HoloLens und den Handcontroller Scans und die finale Transformation erfasst. Über die Punktanzahl ohne VoxelGrid Filter und die maximale Anzahl an Punkten, lässt sich die Raumabdeckung für jeden Testdurchlauf bestimmen. Abbildung 10-8 zeigt die Raumabdeckung im Verlauf der Aufnahme A, Abbildung 10-9 zeigt die Raumabdeckung der Aufnahme B. Die finale Matrix wird in ihre Translation und Rotation Anteile zerlegt. Über die Referenztransformation lässt sich eine Abweichung und darüber auch eine Genauigkeit feststellen.

Getestet werden wie schon in Kapitel 10.3 die Implementationen des ICP, ICP-NL, ICP-Plane und GICP. Ebenfalls wird die maximale Distanz zwischen zwei Punktpaaren auf den Wert ein Meter festgelegt und $\tau = 10^{-10}$ gesetzt. Die maximale Distanz zwischen zwei Punktpaaren könnte die benötigte Zeit zum Ausrichten beeinflussen. Um das Testvolumen zu verringern, wird dieser Parameter jedoch nicht getestet. Wie schon erwähnt gibt es zwei unterschiedliche Scanaufnahmen, die zum Testen genutzt werden. Um auch die Vorausrichtung per Gravitation und Kompass zu testen, werden die Tests sowohl mit der Translations- und Rotationsvorausrichtung, als auch lediglich mit einer Rotationsvorausrichtung durchgeführt.

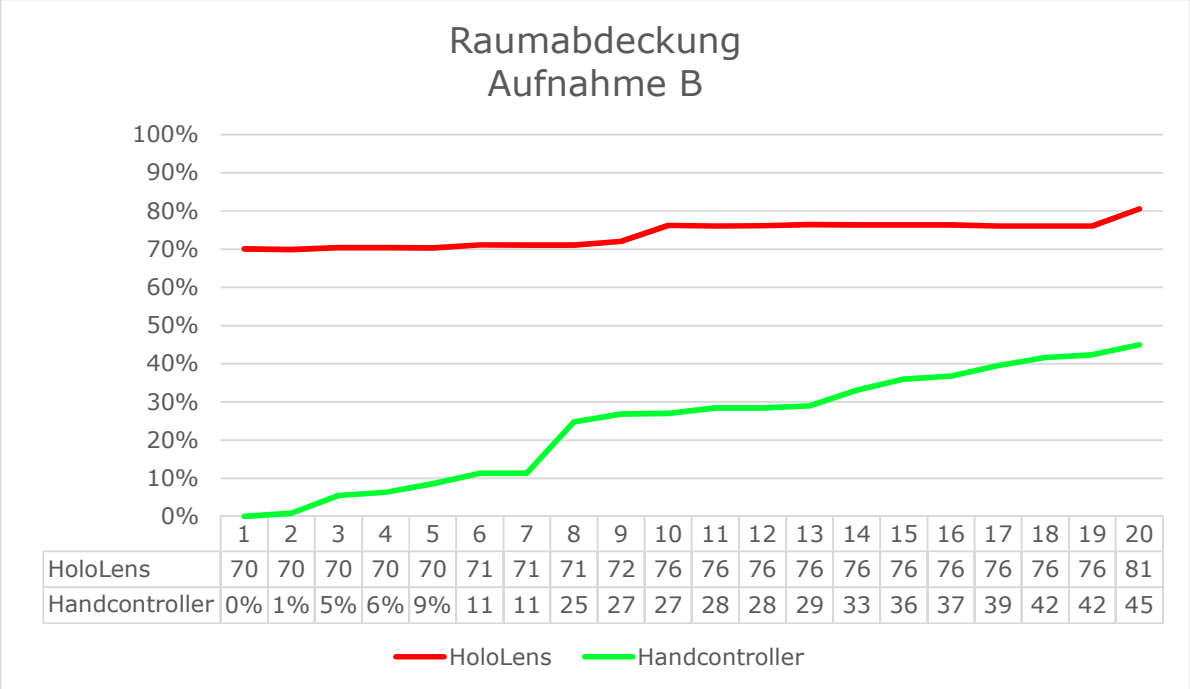
Ursprünglich sollte zu den originalen Punktwolken auch die Verwendung eines VoxelGrid Filter mit den Würfelgrößen 0,5, 1, 2, 5, 10, 20 und 50 Centimetern getestet werden. Da sich die Punktanzahl sowohl bei HoloLens als auch bei dem Handcontroller zwischen den Größen 0,5, 1 und 2 Centimetern kaum unterscheiden und um das Testvolumen weiter zu verringern, werden die Größen 0,5 und 2 Centimeter ausgelassen.

Getestet wird auf einem Windows 10 Computer mit einem Intel Core i7-4712HQ CPU. Dieser besitzt Taktrate von 2,30 Gigahertz.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
HoloLens	63	63	64	64	65	64	64	65	64	65	64	66	66	66	67	70	70	69	70	70
Handcontroller	5%	6%	9%	15	16	21	25	33	33	36	38	42	47	54	56	58	60	60	64	65

Abbildung 10-8: Raumabdeckung Aufnahme A



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
HoloLens	70	70	70	70	70	71	71	71	72	76	76	76	76	76	76	76	76	76	76	81
Handcontroller	0%	1%	5%	6%	9%	11	11	25	27	27	28	28	29	33	36	37	39	42	42	45

Abbildung 10-9: Raumabdeckung Aufnahme B

10.4.2 Vorausrichtung mit Rotation und Translation

Abbildung 10-10 und Abbildung 10-11 zeigen die Punktwolken der Aufnahmen A und B mit der jeweiligen Vorausrichtung. Die Vorausrichtung der Aufnahme A weist eine hohe Rotationsabweichung auf. Aufnahme B benötigt lediglich eine kleine Rotations- und Translationsanpassung.

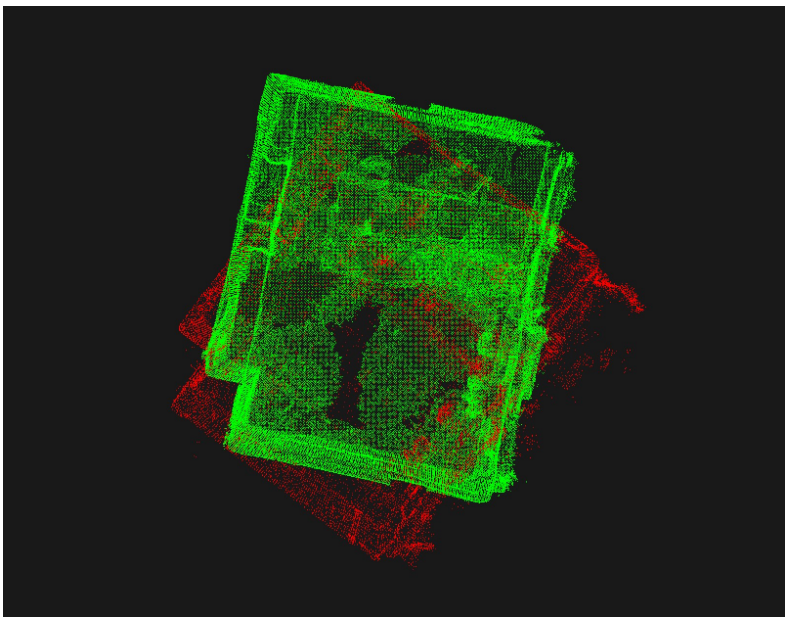


Abbildung 10-10: HoloLens Punktwolke (Rot) und Handcontroller Punktwolke (Grün) der Aufnahme A mit Vorausrichtung

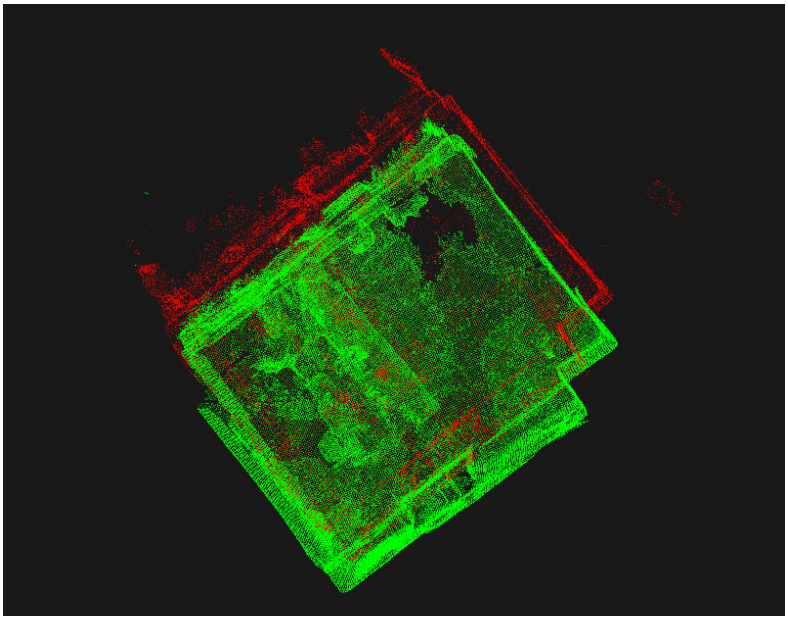


Abbildung 10-11: HoloLens Punktwolke (Rot) und Handcontroller Punktwolke (Grün) der Aufnahme B mit Vorausrichtung

Abbildung 10-12 und Abbildung 10-13 zeigen die Abweichung der Translation von der Referenztransformation bei Aufnahme A an. Die Angaben der Y-Achse sind in Metern. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Zunächst erreicht der GICP bei einer Raumabdeckung von 15% eine Genauigkeit von 10 Centimetern. Der ICP-Plane erreicht mit einer Raumabdeckung von 21% eine bessere Genauigkeit. Beide Algorithmen verbessern sich mit zunehmender Raumdeckung. Der ICP und der ICP-NL benötigen zunächst eine höhere Raumabdeckung, um eine ähnliche Genauigkeit zu erreichen. Beide weisen einen ähnlichen Verlauf auf. Interessant ist, dass nach zunehmender Raumabdeckung, beide zunächst an Genauigkeit verlieren.

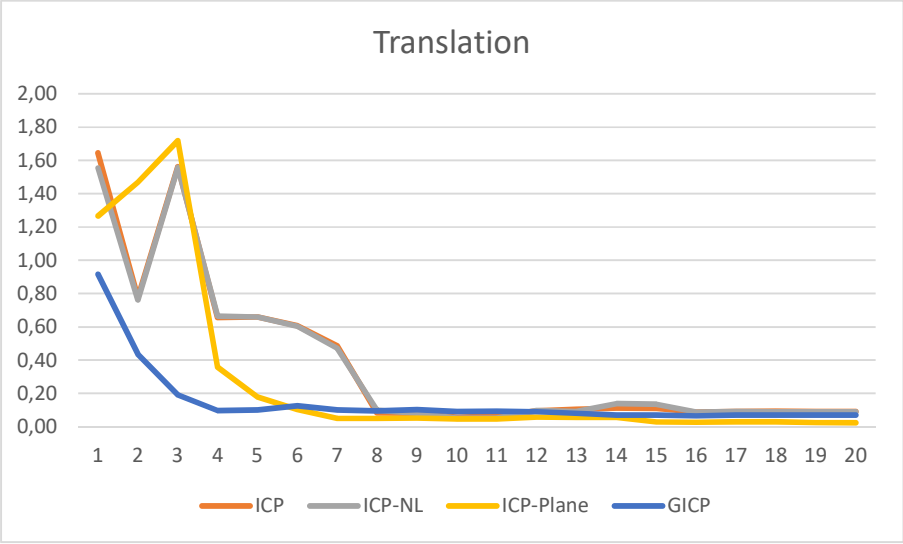


Abbildung 10-12: Translationsabweichung der Aufnahme A ohne VoxelGrid Filter

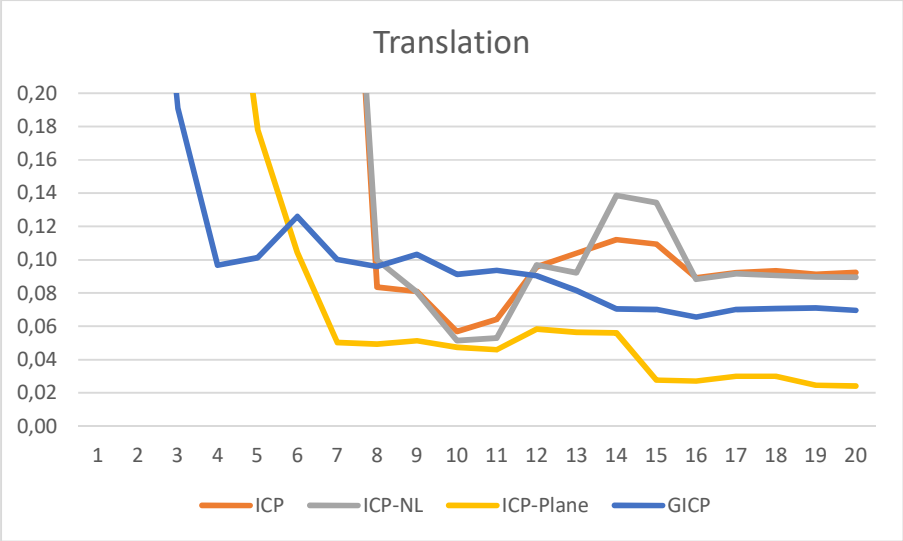


Abbildung 10-13: Translationsabweichung der Aufnahme A ohne VoxelGrid Filter in einer weiteren Skalierung

Abbildung 10-14 und Abbildung 10-15 zeigen die Abweichung der Rotation von der Referenztransformation bei Aufnahme A an. Die Angabe der Y-Achse ist in Grad. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Wie bei der Translation erreicht der GICP bei einer Raum Abdeckung von 15% zunächst die beste Genauigkeit. Der ICP-Plane benötigt dagegen eine Raumabdeckung von 21% um ein ähnliches Ergebnis vorzuweisen. Beide Algorithmen verbessern sich mit zunehmender Raumabdeckung. Ab einer Raumabdeckung von 58% liefert der ICP-Plane eine bessere Genauigkeit als GICP. Der ICP und der ICP-NL weisen ebenfalls einen sehr ähnlichen Verlauf auf. Erst ab einer Raumabdeckung von 58% weisen diese eine konstante Genauigkeit auf, die jedoch schlechter als die des ICP-Plane oder GICP ist.

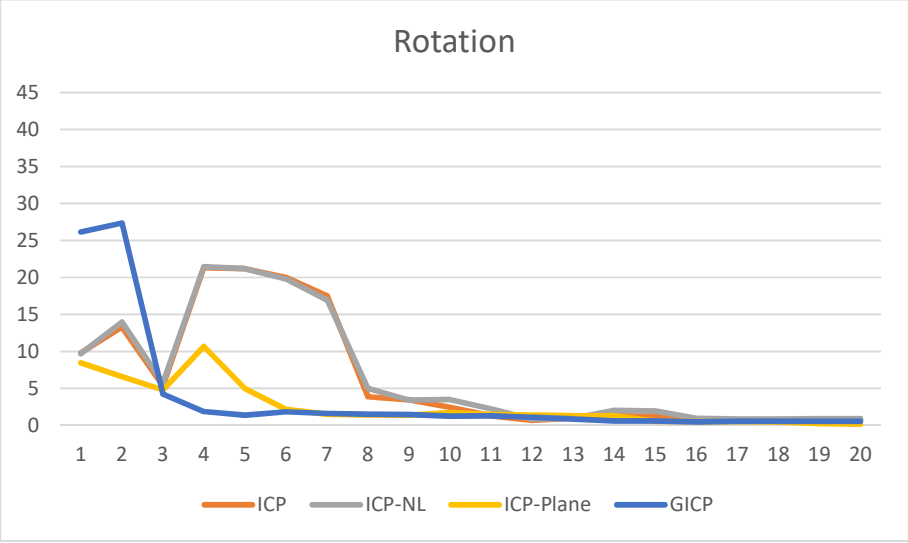


Abbildung 10-14: Rotationsabweichung der Aufnahme A ohne VoxelGrid Filter

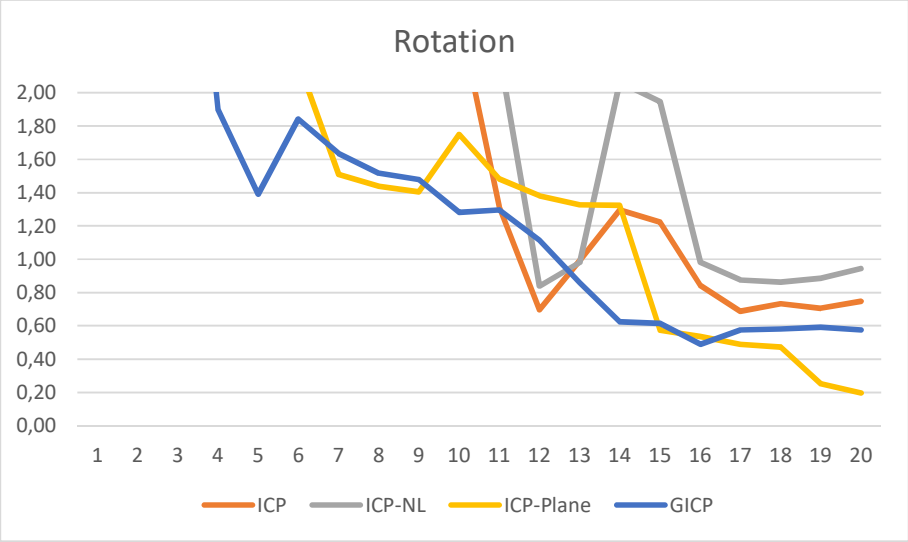


Abbildung 10-15: Rotationsabweichung der Aufnahme A ohne VoxelGrid Filter in einer weiteren Skalierung

Abbildung 10-16 und Abbildung 10-17 zeigen die Abweichung der Translation von der Referenztransformation bei Aufnahme B an. Die Angaben der Y-Achse sind in Metern. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Bei der Aufnahme B liefert der ICP bereits mit einer Raumabdeckung von 1% eine sehr gute Genauigkeit auf. Diese nimmt jedoch mit zunehmender Raumabdeckung wieder ab. Der ICP-NL und ICP-Plane erreichen mit einer Raumabdeckung von 5% und der GICP mit einer Raumabdeckung von 6% als nächste eine ähnliche Genauigkeit. Während der ICP-Plane und der GICP zunächst eine konstante Genauigkeit aufweisen und sich ab 37% verbessern, weichen der ICP und der ICP-NL zunächst wieder ab und erreichen lediglich eine leicht schlechtere Genauigkeit als der ICP-Plane und GICP.

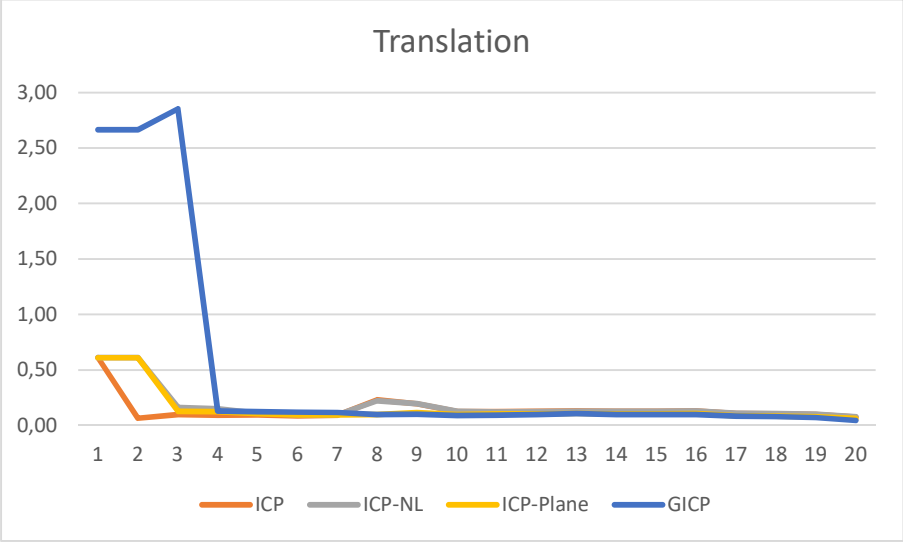


Abbildung 10-16: Translationsabweichung der Aufnahme B ohne VoxelGrid Filter

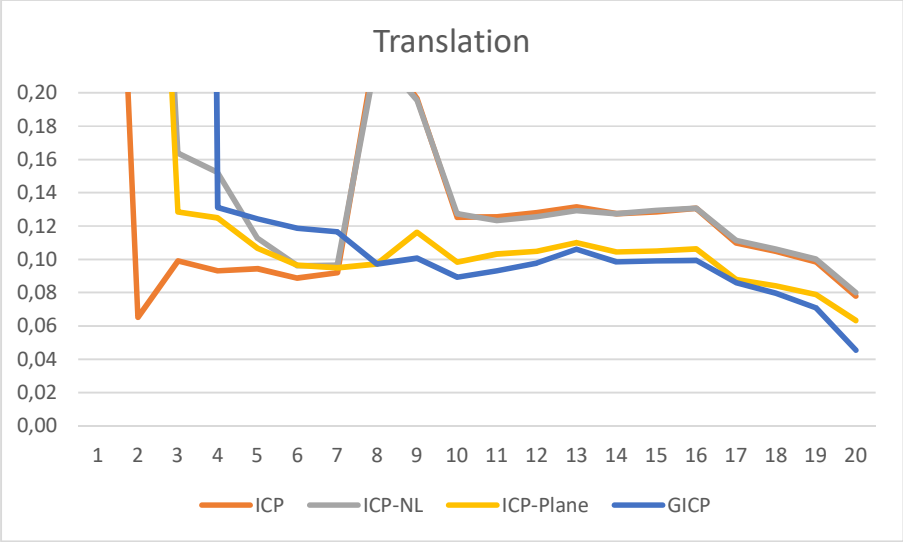


Abbildung 10-17: Translationsabweichung der Aufnahme B ohne VoxelGrid Filter in einer weiteren Skalierung

Abbildung 10-18 und Abbildung 10-19 zeigen die Abweichung der Rotation von der Referenztransformation bei Aufnahme B an. Die Angabe der Y-Achse ist in Grad. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Bei der Rotation beginnen alle zunächst mit einer ähnlichen Abweichung. Wie auch bei der Translation verschlechtern sich der ICP und ICP-NL zunächst bei Sekunde 7 der Aufnahme. Auch bei dem ICP-Plane und GICP erkennt man zunächst eine leichte Verschlechterung. Ab einer Raumabdeckung von 42% verbessert sich die Genauigkeit schlagartig. Der ICP-Plane und GICP weisen eine ähnliche Genauigkeit auf, wobei der ICP-Plane etwas besser ist.

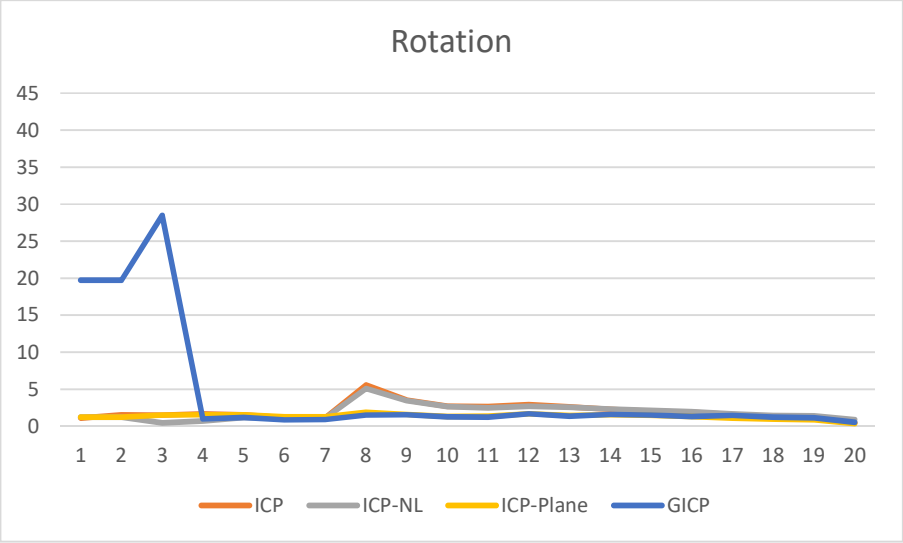


Abbildung 10-18: Rotationsabweichung der Aufnahme B ohne VoxelGrid Filter

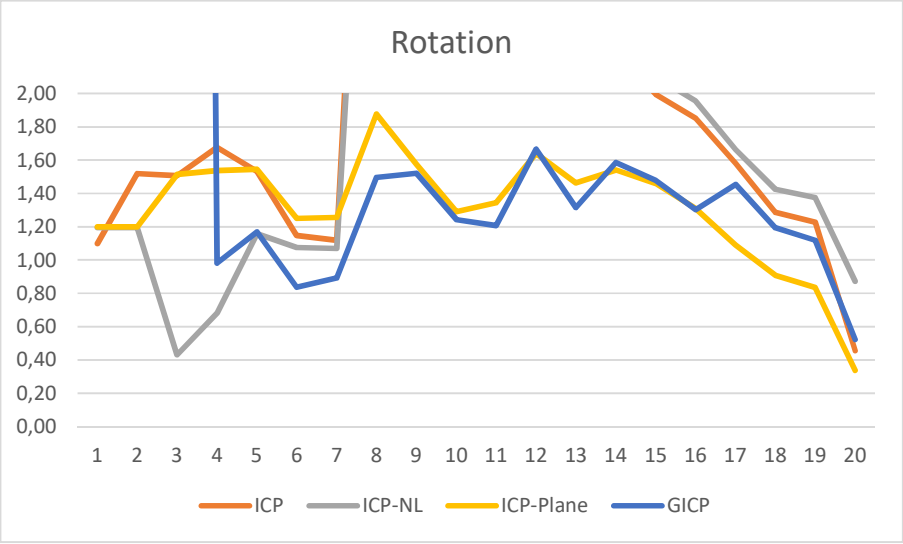


Abbildung 10-19: Rotationsabweichung der Aufnahme B ohne VoxelGrid Filter in einer weiteren Skalierung

Der ICP-Plane und GICP liefern nach geringerer Raumabdeckung bereits bessere Genauigkeiten, als der ICP und ICP-NL. Der ICP und ICP-NL verschlechtern sich teilweise zunächst mit zunehmender Raumabdeckung. Der ICP-Plane und GICP verbessern sich lediglich oder fluktuieren nur leicht. Bei der Aufnahme A benötigen der ICP-Plane und GICP 56% Raumabdeckung, um eine konstante oder sich leicht verbessernde Genauigkeit vorzuweisen. Bei der Aufnahme B sieht man ab einer Raumdeckung von 45% eine entsprechend gute Genauigkeit. Somit wird eine ungefähre Mindestraumabdeckung von 45 bis 55% benötigt, um eine entsprechende Genauigkeit zu erreichen.

Abbildung 10-20 und Abbildung 10-21 zeigen die Laufzeiten der Algorithmen für die Aufnahme A und B an. Die Y-Achse ist in Millisekunden. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Auch bei der Laufzeit schneiden der ICP-Plane und der GICP bei Fall A sehr viel besser als der ICP und ICP-NL ab. Der GICP ist bis zu einer Raumabdeckung von 47% unter dem Zeitlimit. Der ICP-Plane ist sogar bis zu einer Raumabdeckung von 60% unter dem Zeitlimit. Der ICP ist dabei am langsamsten und der ICP-Plane am schnellsten. Bei Fall B sind alle Algorithmen relativ gleich schnell. Der ICP und GICP sind bis zu einer Raumabdeckung von 37% unter dem Zeitlimit und der ICP-NL und ICP-Plane bis zu einer Raumabdeckung von 42%.

Bei der Aufnahme A ist somit sowohl mit dem ICP-Plane und dem GICP ohne einen VoxelGrid Filter die Anforderung an das Zeitlimit erreicht. Bei der Aufnahme B ist das Problem, dass erst nach über 20 Sekunden Scannen eine Raumabdeckung von 50% erreicht wird. Anderenfalls wäre es vermutlich auch dort möglich das Zeitlimit einzuhalten. Es können somit sowohl der ICP-Plane als auch der GICP empfohlen werden, wobei der ICP-Plane teilweise schneller ist und eine bessere Genauigkeit liefert.

Dennoch werden auch die Ergebnisse der Test mit den VoxelGrid Filtern evaluiert. Bei der Aufnahme A liefern die Größen 1, 5 und 10 Centimeter sehr ähnliche Ergebnisse. Interessant ist das bei der Größe von einem Centimeter der ICP und ICP-NL eine sehr viel langsamere Laufzeit aufweisen. Bei der Größe von 20 Centimetern liefert der GICP weiterhin gute Ergebnisse. Der ICP-Plane benötigt jedoch eine Raumabdeckung von 60% um zu ähnlichen Ergebnissen zu kommen. Bei einer Größe von 50 Centimetern erreichen der ICP und der ICP-NL bei der Translation weiterhin ähnliche Ergebnisse. Ab einer Raumabdeckung von 58% produzieren der ICP-Plane und GICP Ausreißer und liefern ein schlechteres Ergebnis als bei kleineren Größen. Bei der Aufnahme B liefern die Größen 1, 5 und 10 Centimeter ebenfalls sehr ähnliche Ergebnisse. Bei der Größe von 20cm beginnt der ICP-Plane je nach Raumabdeckung zu fluktuieren und erreich eine schlechtere Genauigkeit. Weiterhin weist der ICP eine langsamere Laufzeit auf. Bei der Größe von 50 Centimetern entstehen bei der Rotation Fluktuationen und die Algorithmen weisen eine schlechtere Genauigkeit auf. Somit kann ein VoxelGrid Filter bis zu einer Größe von 10 Centimetern empfohlen werden. Dies ermöglicht es allen Algorithmen innerhalb des Zeitlimits eine Konvergenz zu erreichen.

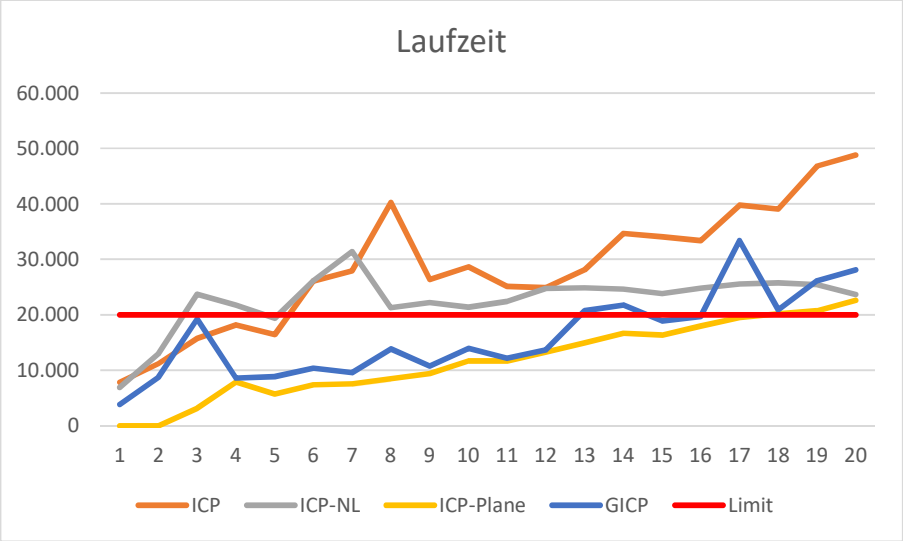


Abbildung 10-20: Laufzeit der Algorithmen bei Aufnahme A ohne VoxelGrid Filter

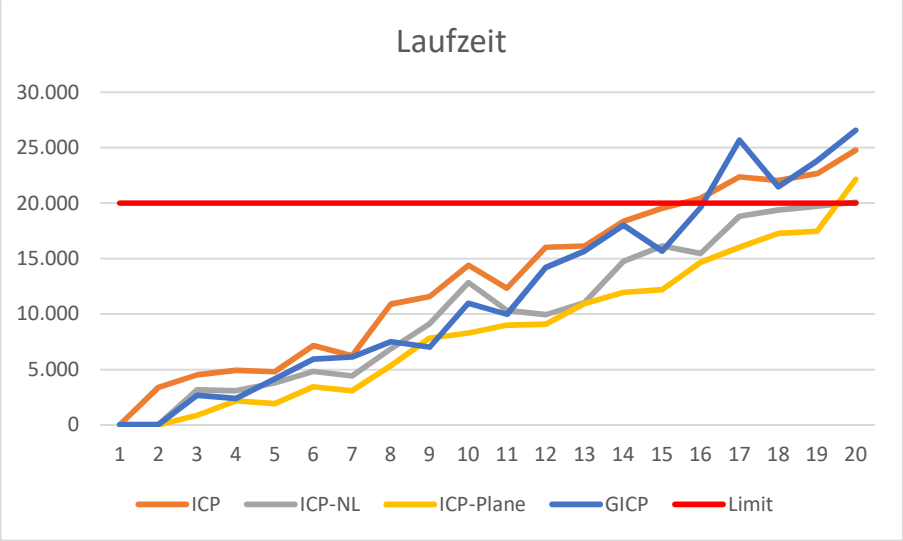


Abbildung 10-21: Laufzeit der Algorithmen bei Aufnahme B ohne VoxelGrid Filter

10.4.3 Vorausrichtung nur mit Rotation

Dieser Test soll das ursprüngliche Vorausrichtung per Kompass simulieren. Dazu wird lediglich der Rotationsanteil zur Vorausrichtung verwendet. In Abbildung 10-22 und Abbildung 10-23 sind die Vorausrichtungen der Aufnahme A und B zusehen. Die Aufnahme A weist lediglich eine kleine Verschiebung auf. Bei der Aufnahme B ist hingegen ein großer Versatz festzustellen.

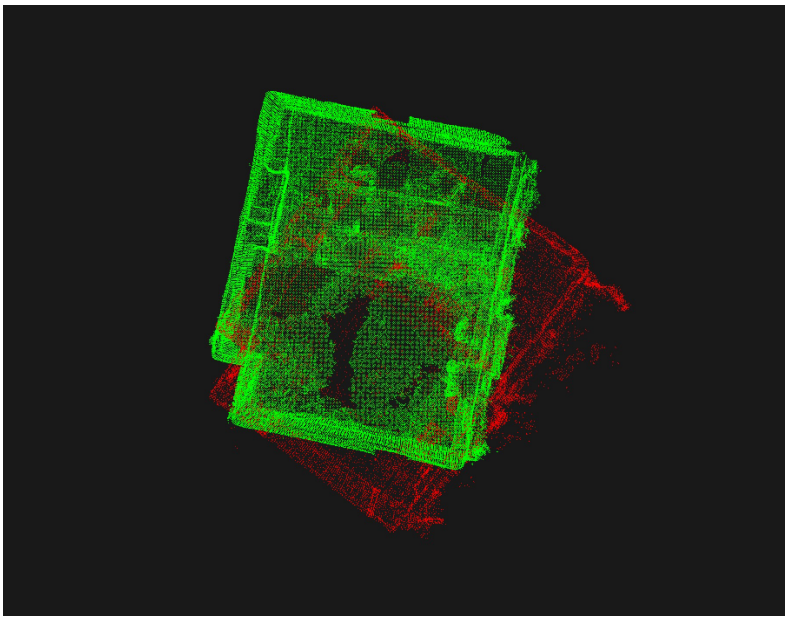


Abbildung 10-22: HoloLens Punktwolke (Rot) und Handcontroller Punktwolke (Grün) der Aufnahme A nur mit Rotationsvorausrichtung

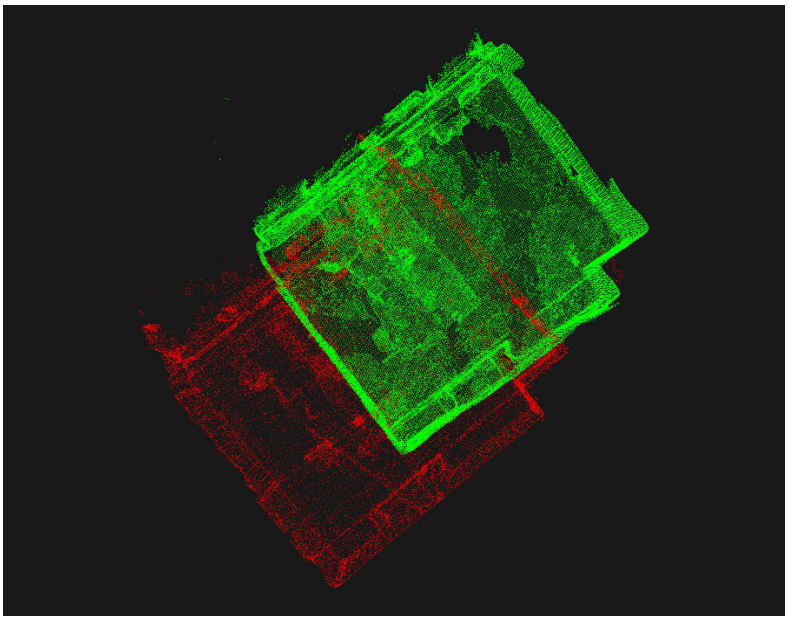


Abbildung 10-23: HoloLens Punktwolke (Rot) und Handcontroller Punktwolke (Grün) der Aufnahme B nur mit Rotationsvorausrichtung

Die Aufnahme A liefert für die Translation und Rotation ähnlich bis gleiche Ergebnisse wie bei den Ergebnissen der Vorausrichtung mit Translation. Die Translation der Vorausrichtung beträgt lediglich 0,55 Meter. Abbildung 10-24 zeigt die Laufzeiten der Algorithmen für die Aufnahme A. Die Y-Achse ist in Millisekunden. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Bei der Laufzeit sind der ICP und der ICP-NL schneller als bei der Vorausrichtung mit Translation. Der ICP-Plane ist sogar sehr viel schneller. Lediglich der GICP benötigt im Vergleich etwas mehr Zeit.

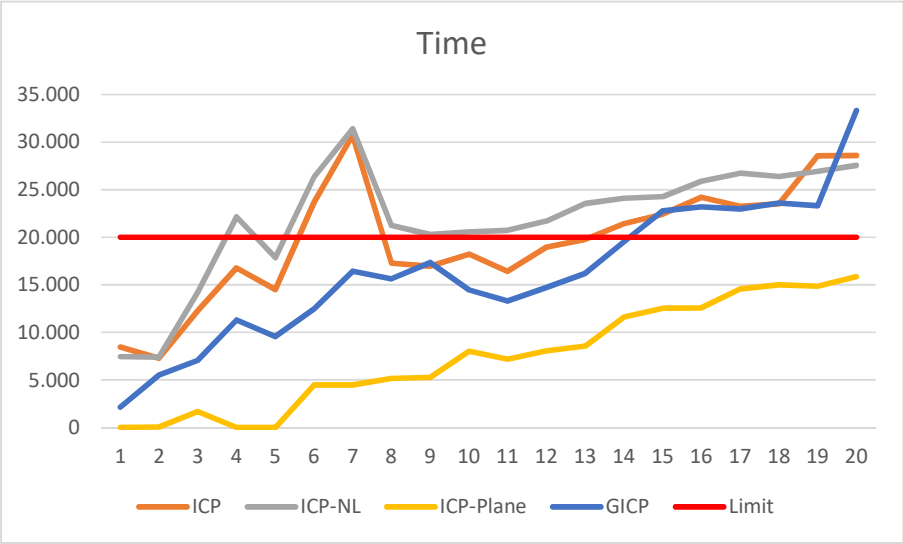


Abbildung 10-24: Laufzeit der Algorithmen bei Aufnahme A ohne VoxelGrid Filter

Abbildung 10-26 und Abbildung 10-25 zeigen die Abweichung der Translation von der Referenztransformation bei Aufnahme B an. Die Angaben der Y-Achse sind in Metern. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Aufnahme B resultiert in einem anderen Ergebnis als die Vorausrichtung mit Translation. Der ICP, ICP-NL und GICP weisen eine ähnliche Genauigkeit in abhängig der Raumabdeckung auf. Erst ab einer Raumabdeckung von 33% bis 36% nimmt die Genauigkeit maßgeblich zu. Der GICP ist dennoch etwas genauer. Der ICP-Plane fluktuiert sehr stark, erreicht aber auch teilweise eine ähnliche Genauigkeit wie der GICP.

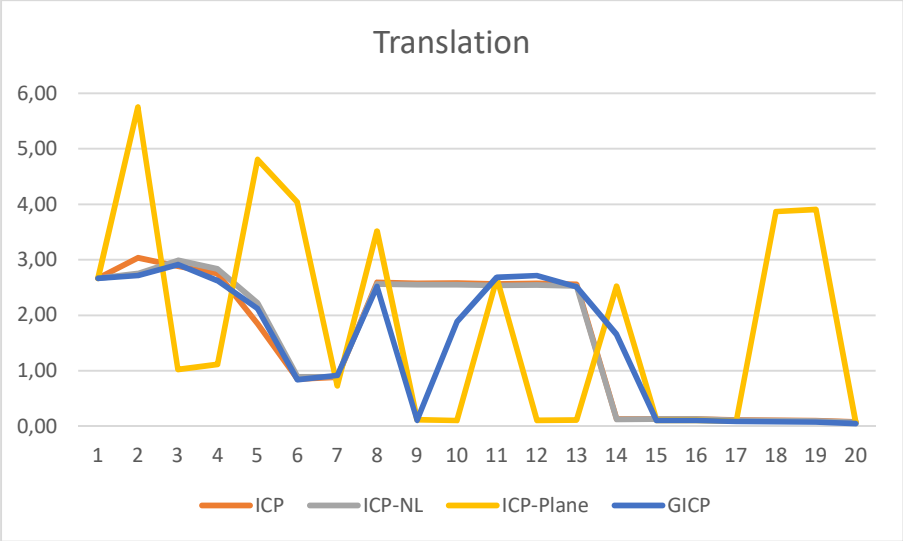


Abbildung 10-26: Translationsabweichung der Aufnahme B ohne VoxelGrid Filter

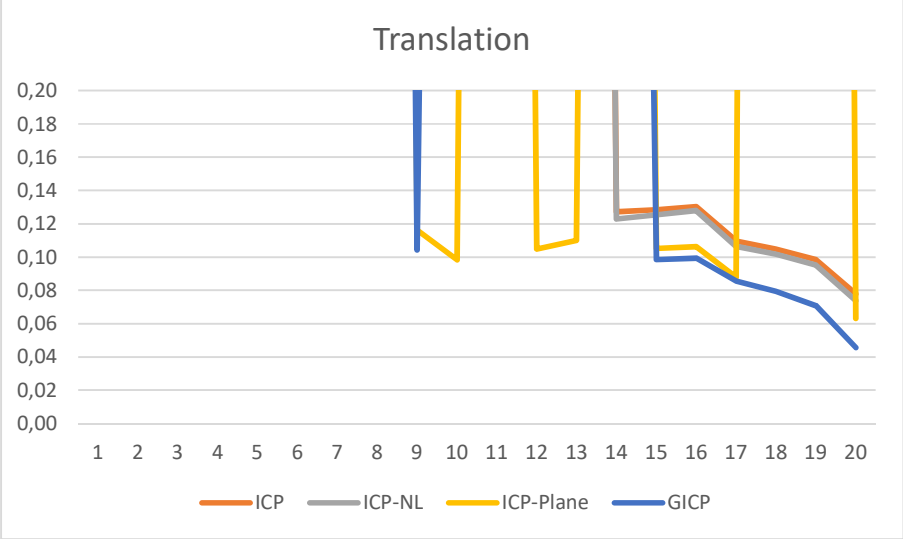


Abbildung 10-25: Translationsabweichung der Aufnahme B ohne VoxelGrid Filter in einer weiteren Skalierung

Abbildung 10-27 und Abbildung 10-28 zeigen die Abweichung der Rotation von der Referenztransformation bei Aufnahme B an. Die Angabe der Y-Achse ist in Grad. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Auch bei der Rotation erreichen der ICP, ICP-NL und GICP erst mit einer Raumabdeckung von 33% bis 36% eine vergleichbare Genauigkeit, die sich mit steigende Raumabdeckung verbessert. Wie auch bei der Translation weist der ICP-Plane hohe Fluktuationen auf, erreicht aber teilweise ebenfalls ähnliche Genauigkeiten, wie die anderen Algorithmen.

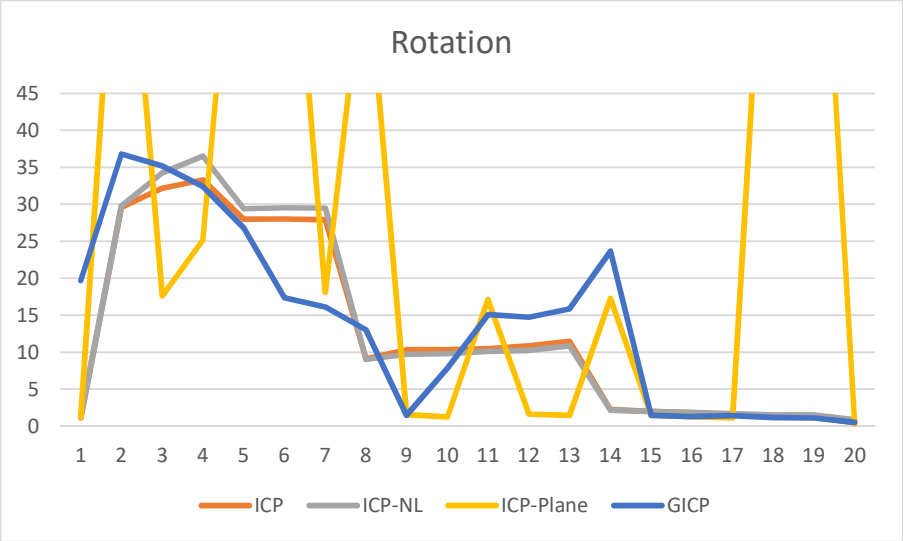


Abbildung 10-27: Rotationsabweichung der Aufnahme B ohne VoxelGrid Filter

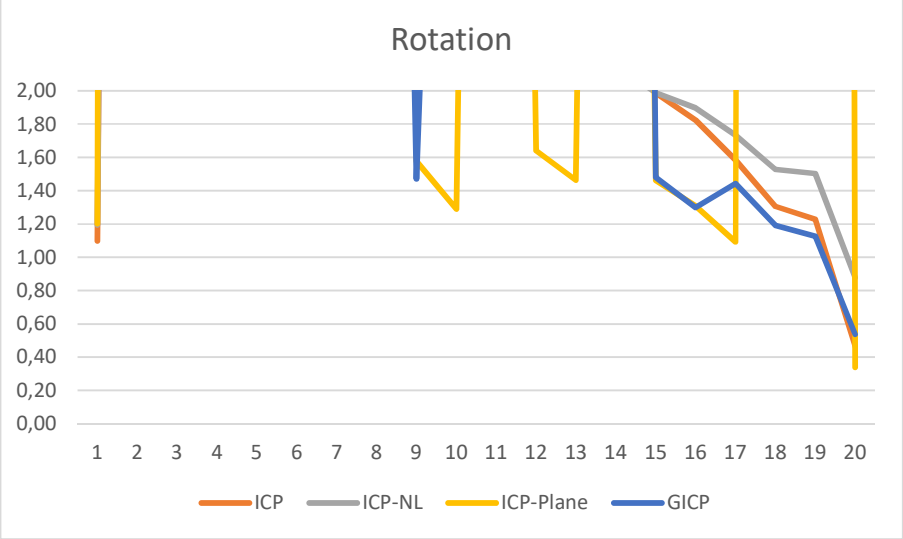


Abbildung 10-28: Rotationsabweichung der Aufnahme B ohne VoxelGrid Filter in einer weiteren Skalierung

Abbildung 10-29 zeigt die Laufzeiten der Algorithmen für die Aufnahme B. Die Y-Achse ist in Millisekunden. Die X-Achse weist den Testdurchlauf mit den abgespielten Sekunden auf. Es ist zu erkennen, dass alle Algorithmen bereits mit einer Raumabdeckung von 6% bis 11% das Zeitlimit überschreiten. Der ICP-Plane benötigt zwar bei einer höheren Raumabdeckung zunächst wieder weniger Zeit, übersteigt dies aber endgültig ab einer Raumabdeckung von 28%.

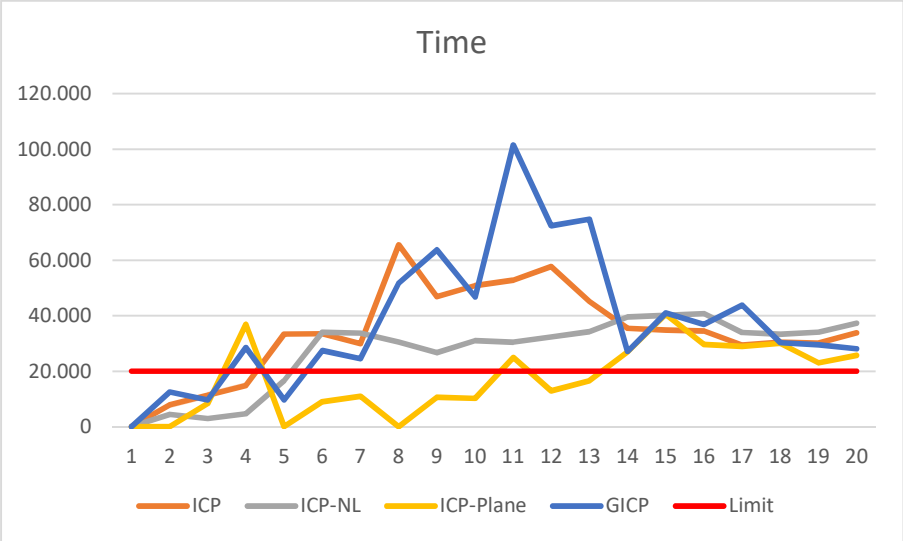


Abbildung 10-29: Laufzeit der Algorithmen bei Aufnahme B ohne VoxelGrid Filter

Der VoxelGrid Filter mit den Größen von 1 und 5 Centimetern weist ähnliche Ergebnisse auf, wie beim Verwenden von keinem Filter. Dennoch überschreiten der ICP, ICP-NL und GICP ab einer Raumabdeckung von 25% das Zeitlimit. Der ICP-Plane überschreitet ebenfalls ab einer Raumabdeckung von 29% das Zeitlimit. Bei den Größen 10 und 20 Centimetern konvergiert der ICP-Plane durchgehend zu einer falschen Ausrichtung. Der GICP weist ebenfalls Falschausrichtungen auf, konvergiert aber dennoch vereinzelt zur richtigen Ausrichtung. Der ICP und ICP-NL erreichen beide eine ähnliche Genauigkeit. Ab einer Raumabdeckung von 39% erreichen beide zunächst eine konstante Genauigkeit, die ab einer Raumabdeckung von 45% weiter verbessert wird. Bei einer Größe von 50 Centimetern liefern der ICP und ICP-NL ähnliche Ergebnisse wie bei 20 Centimetern. Der ICP-Plane fluktuiert weiterhin. Lediglich der GICP weist nur noch eine einzelne Falschausrichtung auf und liefert eine höhere Genauigkeit als der ICP und ICP-NL.

Da die Aufnahme A kaum unterschiedliche Ergebnisse zu der Vorausrichtung mit Translation liefert, wird nur Aufnahme B zur Auswertung betrachtet. Sollte kein VoxelGrid Filter verwendet werden, empfiehlt es sich den GICP zu verwenden, da der ICP-Plane Falschausrichtungen aufweist. Bei der Verwendung eines Filters, können aufgrund der Falschausrichtungen sowohl der ICP-Plane als auch der GICP ausgeschlossen werden. Daher sollte der ICP oder ICP-NL verwendet werden. Unabhängig davon überschreiten alle Algorithmen ab einer Raumabdeckung von 28% das Zeitlimit. Daher kann kein Algorithmus empfohlen werden. Stattdessen sollte eine entsprechende Vorausrichtung mit Translation vorgenommen werden.

11. Fazit und Ausblick

Das vorgestellte Tracking System stellt eine neuartige Alternative zu den herkömmlichen Verfahren dar. Es erfüllt die Anforderungen, dass es ohne externe Basisstation funktioniert und in jedem beliebigen Innenraum verwendbar ist. Auch das Zeitlimit für die Initialisierungsphase ist mit der Abwandlung für die Vorausrichtung zu erreichen. Lediglich die geforderte Genauigkeit kann das System in der derzeitigen Form nicht erfüllen. Wie bereits erwähnt ist es jedoch fraglich, ob diese Ungenauigkeit für das Benutzen ein Problem darstellt. Weiterhin ist ungewiss, warum die geforderte Genauigkeit nicht erreicht werden kann. Eine Vermutung wäre, dass die verwendeten SLAM Systeme mobile Geräte sind, die selbst nicht die geforderte Genauigkeit vorzeigen können.

Die verwendete Handcontroller Hardware, das Lenovo Phab 2 Pro, ist als einfacher Prototyp ausreichend, weist jedoch einige Mängel auf. Durch schnelle Bewegungen entstehen Abweichungen, in bestimmten Situationen bricht das Tracking ab und eine ausreichende Driftkorrektur und Relokalisierung ist nicht gegeben. Diese Mängel sollten für ein stabileres System ausgebessert werden.

Die Tests zeigen, dass das ursprüngliche Konzept, lediglich die Kompassrichtung zur Vorausrichtung zu nutzen, nicht ausreichend ist. Eine entsprechend gute Vorausrichtung mit Translation und Rotation ist maßgebend dafür, dass eine korrekte Ausrichtung getroffen und das Zeitlimit eingehalten wird. Somit müssen weitere Verfahren für die Vorausrichtung erarbeitet werden. Eine Möglichkeit wäre die Ausrichtung per Sichtkontakt herzustellen und die Raum Ausrichtung lediglich zur Ermittlung der Maßstabsskalierung einzusetzen.

Trotz dieser Mängel, kann das Verfahren als Grundlage genutzt werden. Für die meisten Mängel wurden bereits Lösungsvorschläge erbracht, die es nun zu erproben gilt. Sind die Mängel behoben, besitzt dieses Tracking System einen großen Vorteil gegenüber den herkömmlichen Systemen. Da keine externen Basisstationen benötigt werden, ist das System einfach und überall einsetzbar. Neben dem Anwendungsfall für einen Handcontroller, könnte es auch dazu genutzt werden, um weitere primäre Augmented und Mixed Reality Geräte in einen gemeinsamen Referenzrahmen zu bringen. Dies ermöglicht Szenarien mit mehreren Geräten und mehreren Nutzern, die sich gemeinsam in einem Mixed Reality Raum befinden. Weiterhin wäre es auch denkbar das Verfahren für weitere SLAM Systeme, wie Roboter oder Drohnen einzusetzen, um diese mit der HoloLens zu steuern.

I. Literaturverzeichnis

Arun, K. S., T. S. Huang, und S. D. Blostein. 1987. „Least-Squares Fitting of Two 3-D Point Sets." *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-9 (5): 698-700.

Ballesta, Monica, Arturo Gil, Oscar Reinoso, Miguel Julia, und Luis M. Jimenez. 2010. „Multi-robot map alignment in visual SLAM." *WSEAS TRANSACTIONS on SYSTEMS* 9: 213-222.

Berger, Matthew, Andrea Tagliasacchi, Lee M. Seversky, Pierre Alliez, Joshua A. Levine, Andrei Scharf, und Claudio T. Silva. 2014. „State of the art in surface reconstruction from point clouds." *EUROGRAPHICS star reports* 161-185.

Besl, P.J., und Neil D. McKay. 1992. „A method for registration of 3-D shapes." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 239-256.

Chen, Yang, und Gérard Medioni. 1992. „Object modelling by registration of multiple range images." *Image and Vision Computing* 145-155.

Chong, T. J., X. J. Tang, C. H. Leng, M. Yogeswaran, O. E. Ng, und Y. Z. Chong. 2015. „Sensor Technologies and Simultaneous Localization and Mapping (SLAM)." *Procedia Computer Science.* 174-179.

Curless, Brian, und Marc Levoy. 1996. „A volumetric method for building complex models from range images." *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.* ACM. 303-312.

Dissanayake, G., H. Durrant-Whyte, und T. Bailey. 2000. „A computationally efficient solution to the simultaneous localisation and map building (SLAM) problem." *International Conference on Robotics and Automation.* San Francisco, CA: IEEE. 1009-1014.

Dryanovski, Ivan. 2016. „3D Reconstruction with Tango." *2016 IEEE Hot Chips 28 Symposium (HCS).* Cupertino, CA: IEEE. 1-24.

Google Inc. 2017. *Tango Developer Overview | Tango.* 19. Januar. Zugriff am 04. Oktober 2017. https://developers.google.com/tango/developer-overview.

Goradia, Ishan, Jheel Doshi, und Lakshmi Kurup. 2014. „A Review Paper on Oculus Rift & Project Morpheus." *International Journal of Current Engineering and Technology* 3196-3200.

Greenspan, M., und M. Yurick. 2003. „Approximate k-d tree search for efficient ICP." *Fourth International Conference on 3-D Digital Imaging and Modeling.* Banff, Alta., Canada: IEEE. 442-448.

Ho, Kin Leong, und Paul Newman. 2006. „Loop closure detection in SLAM by combining visual and spatial appearance." *Robotics and Autonomous Systems* 54 (9): 740-749.

Holmdahl, Todd. 2015. *BUILD 2015: A closer look at the Microsoft HoloLens hardware - Microsoft Devices BlogMicrosoft Devices Blog.* 30. April. Zugriff am 11. Oktober 2017. https://blogs.windows.com/devices/2015/04/30/build-2015-a-closer-look-at-the-microsoft-hololens-hardware.

Islam, Shahidul , Bogdan Ionescu , Cristian Gadea, und Dan Ionescu. 2016. „Indoor Positional Tracking Using Dual-Axis Rotating Laser Sweeps." *Instrumentation and Measurement Technology Conference Proceedings.* Taipei: IEEE. 1-6.

Jackson, Simon. 2017. *Windows 10 Mixed Reality Devices – The definitive consumer review | Dark Genesis.* 6. September. Zugriff am 11. September 2017. https://darkgenesis.zenithmoon.com/windows-10-mixed-reality-devices-the-definitive-consumer-review/.

Klingensmith, Matthew, Ivan Dryanovski, Siddhartha S. Srinivasa, und Jizhong Xiao. 2015. „CHISEL: Real Time Large Scale 3D Reconstruction Onboard a Mobile Device using Spatially-Hashed Signed Distance Fields." *Robotics: Science and Systems* 4.

Lang, Ben. 2017. *Hands-on: Microsoft's "Mixed Reality" VR Motion Controllers – Road to VR.* 28. August. Zugriff am 11. September 2017. https://www.roadtovr.com/microsoft-mixed-reality-vr-motion-controllers-hands-on/.

Lee, K. H., H. Woo, und T. Suk. 2001. „Data Reduction Methods for Reverse Engineering." *The International Journal of Advanced Manufacturing Technology* 735–743.

Lenovo. 2017. *Lenovo Phab 2 Pro | Augmented Reality Smartphones | Lenovo US.* Zugriff am 13. Oktober 2017. https://www3.lenovo.com/us/en/smart-devices/-lenovo-smartphones/phab-series/Lenovo-Phab-2-Pro/p/WMD00000220.

Leonard, John J., Paul M. Newman, Richard J. Rikoski, José Neira, und Juan D. Tardós. 2001. „Towards Robust Data Association and Feature Modeling for Concurrent Mapping and Localization." *International Symposium on Robotics Research.* Lorne, Australia: Springer-Verlag Berlin Heidelberg . 7-20.

Leonard, John J., und Paul M. Newman. 2003. „Consistent, Convergent, and Constant-Time SLAM." *International Joint Conference on Artifical Intelligence.* Acapulo, Mexico. 1143-1150.

Li, Ce, Haozuo Wei, und Tian Lan. 2016. „Research and Implementation of 3D SLAM Algorithm Based on Kinect Depth Sensor." *International Congress on Image and Signal Processing, BioMedical Engineering and Informatics.* Datong, China: IEEE. 1070-1074.

Linsen, Lars. 2001. „Point cloud representation." Karlsruhe: Universität Karlsruhe.

Microsoft Corporation. 2017. *Environment considerations for HoloLens.* Zugriff am 5. September 2017. https://developer.microsoft.com/de-de/windows/mixed-reality/environment_considerations_for_hololens.

—. 2017. *FAQ.* Zugriff am 5. September 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/faq.

—. 2017. *Hologram stability.* Zugriff am 19. Oktober 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/hologram_stability.

—. 2017. *HoloLens hardware details.* Zugriff am 03. Oktober 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/hololens_hardware_details.

—. 2017. *Immersive headset hardware details.* Zugriff am 11. September 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/immersive_headset_hardware_details.

—. 2017. *Install the tools.* Zugriff am 13. Oktober 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/install_the_tools.

—. 2017. *Mixed reality capture.* Zugriff am 17. Oktober 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/mixed_reality_capture.

—. 2017. *Motion controllers.* Zugriff am 11. September 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/motion_controllers.

—. 2017. *Spatial anchors.* Zugriff am 09. Oktober 2017. https://developer.microsoft.com/en-us/windows/mixed-reality/spatial_anchors.

Miesnieks, Matt. 2017. *Why is ARKit better than the alternatives? – Super Ventures Blog – Medium.* 31. Juli. Zugriff am 2. Oktober 2017. https://medium.com/super-ventures-blog/why-is-arkit-better-than-the-alternatives-af8871889d6a.

Milford, Michael John. 2008. *Robot Navigation from Nature: Simultaneous Localisation, Mapping, and Path Planning Based on Hippocampal Models.* Springer-Verlag Berlin Heidelberg.

Newcombe, Richard A., Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, und Andrew Fitzgibbon. 2011. „KinectFusion: Real-time dense surface mapping and tracking." *2011 10th IEEE International Symposium on Mixed and Augmented Reality.* Basel: IEEE. 127-136.

Nüchter, Andreas. 2009. *3D Robotic Mapping: The Simultaneous Localization and Mapping Problem with Six Degrees of Freedom.* Springer-Verlag Berlin Heidelberg.

Oculus VR, LLC. 2017. *Zubehör | Oculus.* Zugriff am 8. September 2017. https://www.oculus.com/accessories/.

Orts-Escolano, Sergio, Vicente Morel, José García-Rodríguez, und Miguel Cazorla. 2013. „Point cloud data filtering and downsampling using growing neural gas." *The 2013 International Joint Conference on Neural Networks (IJCNN).* Dallas, TX. 1-8.

Peckham, Matt. 2016. *Sony PlayStation VR Expert Explains How The Technology Works | Time.com.* 5. Oktober. Zugriff am 7. September 2017. http://time.com/4520123/playstation-vr-virtual-reality-headset/.

Pradeep, Vivek, Christoph Rhemann, Shahram Izadi, Christopher Zach, Michael Bleyer, und Steven Bathiche. 2013. „MonoFusion: Real-time 3D reconstruction of small scenes with a single web camera." *2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR).* Adelaide, SA: IEEE. 83-88.

Qayyum, Usman, Qaisar Ahsan, und Zahid Mahmood . 2017. „IMU aided RGB-D SLAM ." *International Bhurban Conference on Applied Sciences and Technology .* Islamabad, Pakistan: IEEE. 337-341.

Razer Inc. 2017. *Razer Hydra Portal 2 Bundle Gaming Controller - PC Motion Sensor.* Zugriff am 26. Mai 2017. https://www2.razerzone.com/de-de/gaming-controllers/razer-hydra-portal-2-bundle.

Rothmaier, Gregor. 2012. „Polygonmodellrekonstruktion aus Punktwolken." Stuttgart, 10. Oktober.

Segal, A., D. Haehnel, und S. Thrun. 2009. „Generalized-ICP." *Robotics: science and systems* 2 (4): 435.

Sixense Entertainment. 2014. *STEM System™.* Zugriff am 11. September 2017. http://sixense.com/wireless.

Sony Interactive Entertainment Europe Limited. 2017. *Accessories | PlayStation VR | PlayStation.* Zugriff am 7. September 2017. https://www.playstation.com/en-gb/explore/playstation-vr/accessories/.

Stachniss, Cyrill. 2009. *Robotic Mapping and Exploration.* Springer-Verlag Berlin Heidelberg.

Süße, Herbert, und Erik Rodner. 2014. *Bildverarbeitung und Objekterkennung: Computer Vision in Industrie und Medizin.* Wiesbaden: Springer.

Valve Corporation. 2017. *SteamVR FAQ - SteamVR - Knowledge Base - Steam Support .* Zugriff am 7. September 2017. https://support.steampowered.com/kb_article.php?ref=7770-WRUP-5951.

—. 2017. *Welcome to Steamworks.* Zugriff am 7. September 2017. https://partner.steamgames.com/vrtracking/.

Vander Does, Jesse. 2017. *Extend Your Phone Screen with the HoloLens.* 1. März. Zugriff am 11. September 2017. https://www.afternow.io/2017/03/01/hololens-phone-controller/.

Vander Does, Jesse, Interview geführt von Julian Löhr. 2017. *Question about your turn your phone into a HoloLens controller PoC* (16. Juni).

Xu, Weijun, Rongxin Jiang, und Yaowu Chen. 2012. „Map alignment based on PLICP algorithm for multi-robot SLAM." *2012 IEEE International Symposium on Industrial Electronics.* Hangzhou: IEEE. 926-930.

Zhang, Zhengyou. 1994. „Iterative point matching for registration of free-form curves and surfaces." *International Journal of Computer Vision* 119–152.

II. Abbildungsverzeichnis

III. Tabellenverzeichnis