# Holographische Visualisierung der Leber mit markergestütztem Tracking an einem Phantom

Bachelorarbeit

Antonia Gerdes

Abteilung Translationale Chirurgische Onkologie
NCT Dresden

| | |
|---|---|
| Erstgutachter: | Prof. Dr.-Ing. Stefanie Speidel |
| Zweitgutachter: | Dr.-Ing. Sebastian Bodenstedt |
| Betreuender Mitarbeiter: | Micha Pfeiffer |

Bearbeitungszeit: Juli 08, 2019 – September 23, 2019


Ich versichere hiermit, die vorliegende Arbeit selbstständig angefertigt zu haben. Die verwendeten Hilfsmittel und Quellen sind im Literaturverzeichnis vollständig aufgeführt.

Dresden, den YY Monat, 20XX

# Zusammenfassung

Short description of motivation, methods, results and discussion

# Inhaltsverzeichnis

# 1. Einführung

This section should describe your motivation and general introduction to the topic as well as the goals of your work.

## 1.1 Motivation

-kein hin und her schauen zwischen Patient und Bildschirm -vermeidung von strahlung

## 1.2 Ziele

Getrackt werden soll das Phantom eines Torsos. Es sollen verschiedene Methoden des markerbasierten Trackings implementiert und ein Hologramm der Leber relativ zu den Markern platziert werden. Zu diesen Methoden zählt Vuforia Model Recognition mit einem Trained Model Dataset, bei der die Bauchplatte des Phantoms als Model Target (Marker) fungiert und von Vuforia getrackt wird. Weitere Marker sind Vuforia ImageTargets, ArUco Marker und kreisförmige Farbmarker. Bei diesen drei Methoden werden mehrere Marker auf dem Torso platziert.

Die Methoden sollen mit Hilfe der NDI Polaris evaluiert werden. Dazu soll dem Modell des Torsos eine Reihe an voneinander unterscheidbaren Modellen (farbige Kugeln o.Ä.) hinzugefügt werde, welche sowohl in die Polaris eingepflegt als auch mit der Hololens angezeigt werden. Diese Modelle werden von VersuchsteilnehmerInnen mit dem Pointer der Polaris angefahren. Aus den Positionsdaten des Pointers und den Positionsdaten des in die Polaris eingepflegten Modells lässt sich der Fehler der jeweiligen Methode berechnen.

## 1.3 ...

# 2. Stand der Forschung

Im Folgenden werden verwandte Arbeiten genannt, wobei besonderer Fokus auf Anwendungen mit der Microsoft Hololens gelegt wird. Weiterhin wird auf verschiedene Möglichkeiten der Registrierung, u.a. durch Marker Tracking eingegangen.

## 2.1 Augmented Reality in der Chirurgie

Augmented Reality kann im medizinischen Bereich in vielerlei Hinsicht eingesetzt werden. Großes Potential haben Anwendungen zur präoperativen Planung [1], intraoperativer Visualisierung und zum Training von Chirurgen [2]. Anwendungen mit *Head-Mounted-Displays* (HMD), also Displays, die wie eine Brille auf dem Kopf getragen werden, bieten zusätzlich den Vorteil, dass der Chirurg seine Hände frei nutzen kann. Eine Herausforderung hierbei ist, dass zur Visualisierung die Daten korrekt mit dem Patienten überlagert (*registriert*) werden müssen. Es muss also erreicht werden, dass die wahrgenommene Position in der Szene mit dem computergenerierten Objekt übereinstimmt [3]. Dazu muss die Kamera kalibriert werden.

Azuma hat bereits 1997 Anwendungsfälle für AR im medizinischen Bereich genannt, unter anderem ein Projekt, bei welchem eine Biopsie des Brustgewebes durch AR unterstützt wurde. Die Position des Tumors in einem Phantom wurde visualisiert und die Einstichstelle sowie der Pfad der Nadel im Gewebe bis zum Tumor geführt [2].

Bei *INPRES* (intraoperative presentation of surgical planning and simulation results)[] wurde als HMD die Optische-Durchsicht-Brille Sony Glasstron verwendet.

## 2.2 Marker Tracking in der Chirurgie

Registrierung kann über das Tracken von Markern erreicht werden. Marker sind Objekte, die in der Szene angebracht werden und durch Methoden der Computer Vision im Videostream erkannt werden können. Durch die Position der Marker im Bild lässt sich dann wiederum die Kamerapose ermitteln und so das 3D Objekt an der richtigen Stelle im Raum anzeigen.

Ha und Hong haben ein System zur Unterstützung einer Knochentumorresektion entwickelt. Um den Patienten und die Instrumente zu tracken, wurde ein Tablet Computer mit Kamera genutzt, auf dem auch gleichzeitig die Visualisierung zu sehen war. Als Marker wurden Würfel genutzt, an welchen auf jeder Seite ein planares Bild angebracht war,

ähnlich zu Aruco Markern. Diese Marker wurde am Patienten und an den Instrumenten befestigt, ein externes Tracking System war nicht notwendig [1].

Kenngott et al. [4] haben ein mobiles System entwickelt, welches eine möglichst patientennahe Anwendung von Augmented Reality ermöglicht. So können Aufnahmen des Patienten schnellstmöglich visualisiert werden. Dies ist vor allem in Umfeldern, wo es auf Schnelligkeit ankommt (bspw. in der Notaufnahme) ein großer Vorteil gegenüber des üblichen Vorgehens, dem Anzeigen der Aufnahmen an einem dafür vorgesehenen Arbeitsplatz. Um den Bezugsrahmen für die AR Anwendung zu definieren, wurden 15 strahlenundurchlässige Marker genutzt, welche frei auf dem Patienten positioniert werden konnten. Für den Registrierungsprozess wurden 6 dieser Marker genutzt und farblich markiert. Zum Tracking der Marker und Anzeigen des Hologramms wurde ein handelsübliches Apple iPad genutzt.

[5]

## 2.3 Hololens in der Chirurgie

Die Microsoft Hololens ist ein HMD für Augmented Reality und zurzeit eines der besten HMDs auf dem Markt geeignet für chirurgische Anwendungen [6]. Es ist ein optical see-through display, ermöglicht also dem Chirurgen gleichzeitig die Sicht auf die reale Umgebung ohne eine Zeitverzögerung. Die Hololens ist nicht kabelgebunden und lässt sich über Handgesten und Sprachbefehle steuern, was die Sterilität im Operationssaal nicht gefährdet [6].

Von Pratt et al. [6] wurde eine Hololens App zur Unterstützung der Chirurgen bei einer Operation zur Rekonstruktion von Extremitäten entwickelt (siehe Abb. 2.2). Es wurde festgestellt, dass die Anwendung verlässlicher und weniger zeitaufwändig ist, als die zur Zeit für solche Eingriffe genutzte Doppler-Sonografie. Das Hologramm wurde beim Start der App ausgehend von der Pose der Hololens an einer festgelegten Stelle im Raum gerendert. Die Registrierung wurde vom Chirurgen manuell durch das nutzen der Air-Tap-and-Hold Geste durchgeführt. Über Sprachbefehle oder entsprechende Buttons in einer Toolbar konnte zwischen Translation und Rotation Modus gewechselt werden.

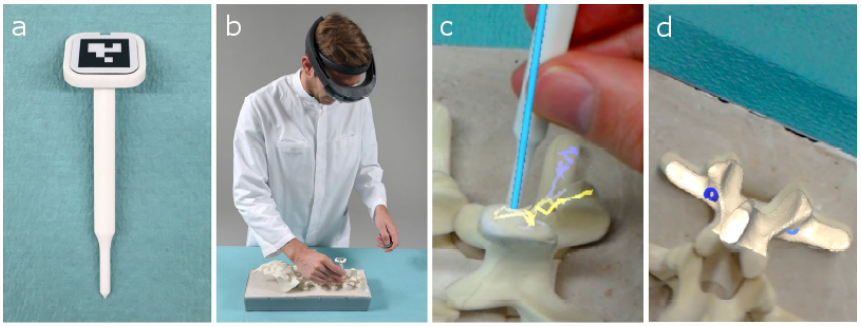


Abb. 2.1: Erstellung der intraoperativen Point Cloud; (**a**)Pointer mit Marker; (**b**)Chirurg mit Hololens; (**c**) Sicht durch Hololens während des Abtastvorgangs; (**d**) Überlagerung nach der Registrierung [7]

Liebmann et al. [7] haben eine Navigationsanwendung für die Platzierung von Pedikelschrauben bei Operationen an der Wirbelsäule entwickelt. Ziel der Anwendung war es, intraoperative Bildgebung wie CT Aufnahmen zu vermeiden, da der Patient dabei einer hohen Strahlenbelastung ausgesetzt wird. Mit 2 der 4 Frontkameras der Hololens auf die über den Research Mode zugegriffen werden kann wurde Marker Tracking zur Bestimmung der Kamerapose umgesetzt. Als Marker wurden sterile kommerziell verfügbare Marker verwendet. Um zu Registrieren, wurde intraoperativ die Oberfläche der Wirbelsäule mit Hilfe

eines Pointers, der getrackt werden kann, abgetastet. Zusätzlich wurde der Klicker der Hololens genutzt. Während der Klicker gedrückt wurde, wurde die Position vom Pointer gespeichert. Die so erhaltene intraoperative Point Cloud wurde anschließend vollautomatisch mit der präoperativen Point Cloud registriert (siehe Abb. 2.1). Abschließend wurde das Hologramm angezeigt und bei korrekter Überlagerung vom Chirurgen bestätigt. Durch doppelklicken des Hololens Klickers wurde dann die Navigation gestartet. Während der Navigation wurde das Hologramm ausgeblendet und nur die Bohrlöcher visualisiert, zusätzlich wurde dem Chirurgen die aktuelle und gewünschte Neigung eines Navigationswerkzeuges angezeigt.

Rae et al. [8] nutzen die Hololens, um die Platzierung von Bohrlöchern auf dem Schädel anzuzeigen. Mit dem Tool 3D Slicer wurde ein 3D Modell erstellt und Marker sowohl an den gewünschten Bohrpunkten als auch an Registrierungspunkten platziert. Die Registrierung des 3D Modells via Hololens mit dem Phantom erfolgte über die manuelle Positionierung des ersten Markers. Die Abstände zwischen den Markern sind fix, sodass die beiden anderen Marker vom Chirurgen an die gewünschte Stelle rotiert werden konnten. Anschließend konnten vom Chirurgen kleinere Anpassungen in der Positionierung vorgenommen werden. Bei dem gesamten Vorgang wurde auf die korrekte Einschätzung des Chirurgen vertraut.

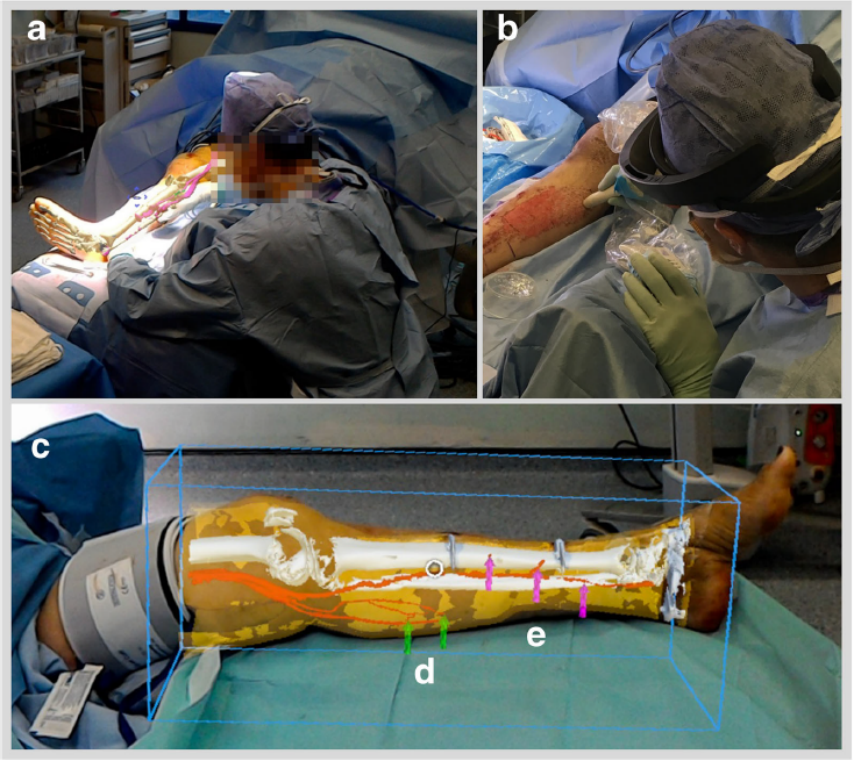


Abb. 2.2: Hololensgestützte Operation; (**a**) Blick auf OP über zweite Hololens; (**b**) Chirurg nutzt Doppler-Sonografie zur Überprüfung; (**c**) Registrierung von Hologramm und Patient, Hervorhebung wichtiger Positionen durch Pfeile [6]

# 3. Grundlagen

In diesem Kapitel wird eine Einführung in die Grundlagen gegeben, welche im Methoden Kapitel verwendet werden. Zuerst wird das Mixed Reality Spektrum erläutert, anschließend das Prinzip der Head Mounted Displays. Im Abschnitt 4.3 wird AR Tracking beschrieben, welches Marker, verschiedene Arten von Markern und Tracking Algorithmen beinhaltet. Abschließend wird das bei dieser Arbeit verwendete Phantom vorgestellt.

## 3.1 Mixed Reality

Der Begriff *Mixed Reality* (MR) wurde zuerst von Milgram und Kishino [9] beschrieben. Dabei handelt es sich um ein Spektrum, welches sich zwischen der realen und der virtuellen Welt aufspannt (siehe Abb. 4.1). Unter der virtuellen Welt wird hierbei die *Virtuelle Realität* (VR) verstanden, welche sich dadurch kennzeichnet, dass der Nutzer vollständig in eine digitalisierte Umgebung eintaucht, ohne seine reale Umwelt direkt wahrnehmen zu können. Auf dem MR Spektrum sind reale und virtuelle Welt unterschiedich stark miteinander verschmolzen. Bei der *Augmented Reality* (AR) werden der realen Welt virtuelle Objekte, wie bspw. der Avatar einer sich nicht im Raum befindlichen Person, hinzugefügt [10]. Bei der *Augmented Virtuality* (AV) werden ausgehend von der virtuellen Welt Eigenschaften des realen Raumes übernommen, wie etwa Hindernisse [10].

Heutzutage wird die MR nicht nur durch die Art des verwendeten Bildschirms beschrieben, sondern zusätzlich durch die Fähigkeit des Geräts, sich im Raum lokalisieren zu können und bspw. Geräusche räumlich korrekt widergeben zu können [10].

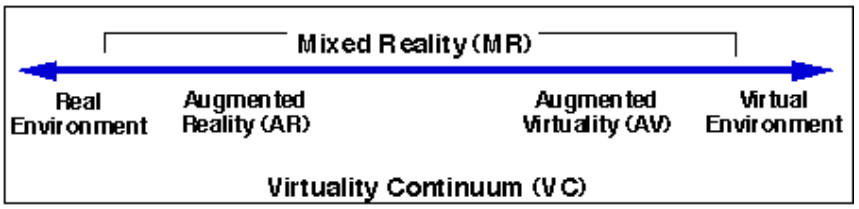


Abb. 3.1: Mixed Reality Spektrum nach Milgram und Kishino [9]

## 3.2 Head Mounted Displays

Um virtuelle Objekte in der realen Welt anzeigen zu können, werden Displays benötigt. Neben Displays, welche in der Hand gehalten werden und stationär befestigten Displays,

gibt es *Head Mounted Displays* (HMDs). Sie werden ähnlich einer Brille auf dem Kopf getragen, das Display befindet sich vor den Augen des Nutzers. Im Bereich AR wird zwischen *video see-through HMD* (VST-HMD) und *optical see-through HMD* (OST-HMD) unterschieden.

Bei VST-HMDs wird ein Video der Umwelt aufgenommen, in welches die virtuellen Objekte mittels Methoden der Computer Vision pixelgenau eingefügt werden [11]. Es tritt keine Zeitverzögerung zwischen Bild der realen Welt und dem darauf angezeigtem virtuellem Objekt auf. Mittels Tiefenkameras kann eine korrekte räumliche Überlagerung von virtuellen Objekten durch reale Objekte garantiert werden. Ein Nachteil von VST-HMDs ist die Tatsache, dass der Träger des HMDs die Welt nur über das Video wahrnimmt und es so zu Problemen aufgrund von begrenzter Auflösung und zeitlicher Verzögerung zwischen realer Welt und Video kommen kann [11].

OST-HMDs hingegen sind Displays, welche durchsichtig sind und so den Blick auf die reale Umgebung ermöglichen. Dadurch gibt es keine Zeitverzögerung zwischen realer Welt und Szene, die der Nutzer sieht, mehr. Allerdings kann es nun zur zeitlichen Verzögerung im Rendering des virtuellen Objektes kommen. Ein weiteres großes Problem ist das Registrieren der virtuellen Objekte mit der Umwelt, da bei OST-HMDs zum positionieren dieser Objekte keine Methoden der Computer Vision verwendet werden können [11]. Die in dieser Arbeit verwendete Microsoft Hololens lässt sich zu den OST-HMDs zählen.

## 3.3 Registrierung

Im medizinischen Bereich ist es wünschenswert, verschiedene Datensätze, welche bspw. durch unterschiedliche Bildgebende Verfahren oder zu verschiedenen Zeitpunkten im Krankheitsverlauf entstanden sind, in Übereinstimmung zu bringen. So können die Datensätze gleichzeitig angezeigt werden, was die Interpretation erleichtert und das aufwändige Wechseln zwischen den verschiedenen Medien vermeidet. Dies ermöglicht eine Verbesserung von Diagnose, Planung und Therapie. Dieser Prozess des in-Übereinstimmung-bringens wird *Registrierung* genannt.

In der AR bezieht sich der Begriff Registrierung auf das Problem, die Objekte der virtuellen und realen Welt miteinander in Einklang zu bringen [2]. Es werden *Ankerpunkte* (Anchor) benötigt, um die Hologramme am korrekten Ort anzuzeigen und dort zu halten, selbst wenn der Nutzer sich im Raum bewegt. Als Anchor können u.a. reale Objekte im Raum, bspw. Image Marker aus Papier, genutzt werden [11]. Nach der initialen Registrierung, bei welcher die Position und Rotation (*Pose*) des Nutzers in Relation zum Anchor bestimmt werden, müssen diese kontinuierlich aktualisiert werden, da der Nutzer sich im Raum bewegt [11]. Im Folgenden werden verschiedene Tracking Methoden genannt.

## 3.4 AR Tracking

Generell gibt es verschiedene Tracking Methoden, wie Magnetisches Tracking, Sichtbasiertes Tracking, Inertial Tracking oder hybride Anwendungen.

Sichtbasiertes Tracking zeichnet sich dadurch aus, dass zur Berechnung der Kamerapose im Verhältnis zu Objekten im Raum Methoden aus der Computer Vision genutzt werden [12]. Da in dieser Arbeit sichtbasiertes Tracking verwendet wurde, wird nur dieses im Weiteren erläutert.

### 3.4.1 Infrarot Tracking

Beim Infrarot Tracking wird von einem Marker entweder Infrarot Strahlung ausgesendet (*aktiver Marker*) oder reflektiert (*passiver Marker*). Für die trackende Kamera haben diese

Marker einen hohen Kontrast, da sie als helle Punkte vor dunklem Hintergrund erscheinen. Es gibt zwei Ansätze, wie Marker und Kamera konfiguriert sein können: als *outside-looking-in* Konfiguration, bei welcher das zu trackende Objekt mit Markern ausgestattet und von einer stationären Kamera getrackt wird oder als *inside-looking-out* Konfiguration, bei welcher das zu trackende Objekt mit einer Kamera ausgestattet wird und sich anhand von im Raum befestigten Markern orientiert [11].

### 3.4.2 Visible Light Tracking

Mit Hilfe von Video Aufnahmen aus RGB-Kameras, wie sie bspw. die Hololens besitzt, lassen sich Hologramme ebenso mit der realen Welt registrieren. Die dazu verwendeten Methoden lassen sich in drei Techniken aufteilen, welche im Folgenden erläutert werden.

#### Fiducial Tracking

*Fiducials* sind phyische Objekte, die in der realen Welt platziert werden (siehe Abb.4.2). Sie können als Anchor für virtuelle Objekte genutzt werden, um diese im Raum zu positionieren. Wie ein Fiducial aussieht, ist stark von dem Algorithmus abhängig, der zum Tracken genutzt wird. So gibt es einfarbige Fiducials, die per Color Matching getrackt werden können [11]. Hierbei muss beachtet werden, dass ein einzelner, einfarbiger Marker nicht genügend Informationen enthält, um die Kamerapose zu bestimmen. Es müssen mindestens vier Punkte im Raum erkannt werden, um diese berechnen zu können [11]. Es muss also darauf geachtet werden, die Marker so in der realen Welt zu positionieren, dass zu jeder Zeit mindestens vier einfarbige Fiducials sichtbar sind. Dieser Mehraufwand kann vermieden werden, indem planare quadratische Marker verwendet werden, deren Ecken als die vier bekannten Punkte dienen. Zusätzlich kann ein Bild dem Marker hinzugefügt werden, um zu ermöglichen, dass mehrere Marker zeitgleich verwendet und voneinander unterschieden werden können, sowie die Rotation um die vertikale Achse feststellen zu können, wie es bspw. beim ARToolkit der Fall ist [11].

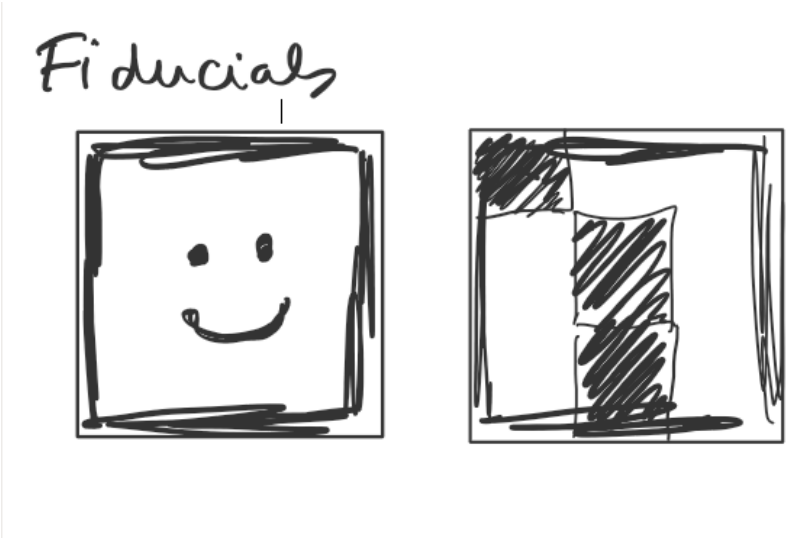


Abb. 3.2: Beispiel planare Fiducials und einfarbige Fiducials (beim zeichnen vergessen)

#### Natural Feature Tracking

Natural Feature Tracking ermöglicht ein markerloses Tracking, somit müssen dem Raum keine künstlichen Merkmale in Form von Fiducials hinzugefügt werden. Algorithmen wie SIFT (Scale Invariant Feature Transform) ermitteln eindeutige Merkmale in einem Set von Referenzbildern und bilden einen Deskriptor, welcher spezifisch für dieses eine Merkmal ist und in einer Datenbank gespeichert wird. Mittels Feature Matching werden Merkmale

des zu trackenden Objekts mit Merkmalen aus der Datenbank abgeglichen. Die Pose der Kamera kann nun mit den gefundenen Features ähnlich berechnet werden wie bei Fiducials [11].

#### Modellbasiertes Tracking

Beim modellbasiertem Tracking wird ein 3D-Modell des Objektes genutzt, welches getrackt werden soll. Auf die Szene werden Kantenfilter angewendet, welche mit mit dem 3D-Modell abgeglichen werden, wodurch die Pose der Kamera bestimmt wird. Es ist möglich, Natural Feature Tracking und die Textur des 3D-Modells mit einzubeziehen, wodurch das Tracking robuster wird [11].

### 3.4.3 3D Structure Tracking

Es ist möglich, mit Hilfe von 3D-Kamerasystemen Tiefendaten zu erhalten. Diese Tiefendaten werden mittels Methoden wie time-of-flight oder strukturiertem Licht erzeugt. Bei ersterem wird die Laufzeit eines Lichtimpulses zum Objekt und zurück gemessen, sodass für jeden Pixel in der Szene eine Laufzeitmessung und somit die Distanz zur Kamera bekannt ist [13]. Bei strukturiertem Licht werden Streifen oder Punkte auf eine Oberfläche projiziert. Durch die Oberfläche wird das projizierte Muster verzerrt. Mithilfe von optischer Triangulation kann nun die 3D Position der Punkte bestimmt werden [13]. Die KinectFusion nutzt strukturiertes Licht um 3D Modelle von Objekten und vom Raum zu erstellen, welche wiederum zum Tracken der Pose der Kamera genutzt werden [11].

## 3.5 Phantom

Unter einem *Phantom* wird in der Medizin die Nachbildung eines Organs verstanden, an welchem Übungen und Experimente durchgeführt werden können. Das in dieser Arbeit verwendete Phantom ist das *open-source Heidelberg laparoscopic phantom* (Open-HELP). Es wurde nach dem CT-Scan des Torsos eines männlichen Patienten modelliert [14]. Der Torso enthält naturgetreue Organe und eine abnehmbare Bauchdecke (siehe Abb. 4.3).

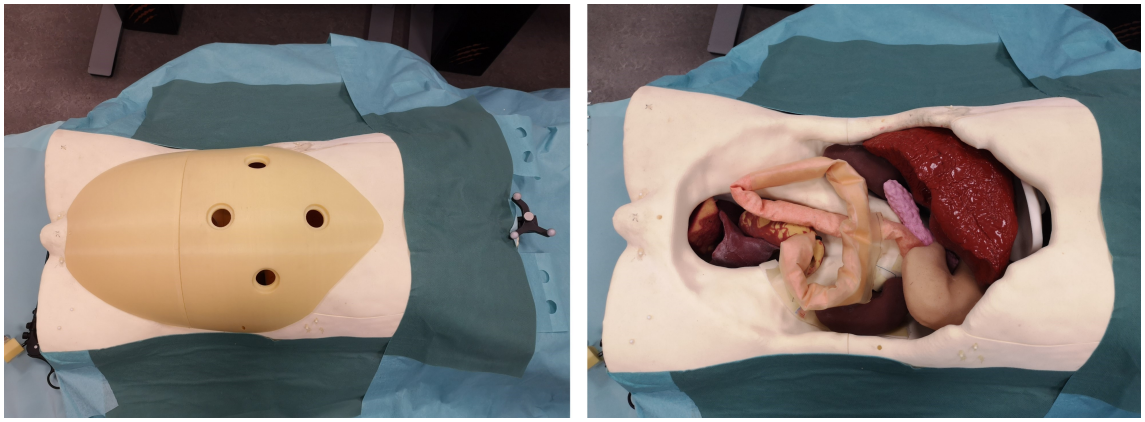

Abb. 3.3: Open-HELP Phantom mit und ohne Bauchdecke

# 4. Methoden

Im Folgenden Kapitel wird auf die in dieser Arbeit genutzte Hardware und SDKs eingegangen, sowie auf die verschiedenen Ansätze zum Versuchsaufbau, die verfolgt wurden. Es wird des Kalibrierungsvorgang erläutert und zuletzt das User Interface, welches genutzt wurde.

## 4.1 Microsoft Hololens

Das in dieser Arbeit genutzte Optical See-Through HMD ist die Hololens der ersten Generation. Die Hololens wurde 2016 von der Microsoft Corporation auf den Markt gebracht. Sie übertrifft andere kommerziell verfügbaren OST-HMDs in Kontrast und Stabilität der Bildwiederholrate und hat eine geringere Zeitverzögerung [15]. Die Hololens ist nicht kabelgebunden, lässt sich über Handgesten sowie Sprachbefehle steuern und gefährdet somit nicht die Sterilität im Operationssaal [6].

Ein virtuelles Objekt, welches von der Hololens angezeigt wird, wird *Hologramm* genannt. Die Hololens projiziert ein Hologramm auf zwei Durchsichtdisplays (*Waveguides)*, für jedes Auge eines. Im Gegensatz zu Video See-Through HMDs, welche bei einem Stromausfall dem Nutzer jegliche Sicht versperren, ermöglichen die Waveguides zeitgleich die Sicht auf die reale Umgebung. Für jeden Waveguide gibt es eine *Light Engine*, welche die holographischen Inhalte projiziert. Mit Hilfe von vier Umgebungssensoren und einer Time-of-Flight Tiefenkamera kann die Hololens eine Karte des Raumes erstellen (*Spatial Mapping*) und sich selbst darin Positionieren. Die Tiefenkamera wird zusätzlich zur Erkennung der Handgesten genutzt. [16]

**Technische Daten**

| Komponente | Daten |
|---|---|
| **Optik** | 2 Durchsichtdisplays (Waveguides)<br>2 HD 16:9 Light Engines |
| **Holographische Auflösung** | 2,3 Millionen Lichtpunkte |
| **Holographische Dichte** | mehr als 2500 Radianten<br>(2500 Lichtpunkte pro Radiant,<br>wobei $1rad \approx 57,3°$ ) |
| **Sensoren** | 1 IMU (Inertiale Messeinheit)<br>4 Umgebungssensoren<br>1 Tiefenkamera (Time-of-Flight)<br>1 2MP HD Kamera<br>4 Mikrofone<br>1 Umgebungslichtsensor |
| **Interaktion** | räumlicher Ton<br>Blickverfolgung<br>Handgesten<br>Sprachsteuerung (bei Internetverbingung) |
| **Leistung** | 2-3 Stunden aktive Nutzung<br>bis zu 2 Wochen standby<br>nutzbar während des Ladevorgangs<br>passive Kühlung |
| **Prozessoren** | Intel 32 bit architecture<br>Microsoft Holographic Processing Unit (HPU 1.0) |
| **Betriebssystem** | Windows 10 |

## 4.2 NDI Polaris Vega

Die Polaris Vega von Northern Digital Inc. ist ein Infrarot Tracking System zur Anwendung im medizinischen Bereich. Die Polaris ist als outside-looking-in System konfiguriert und somit nicht sehr mobil. Die zu trackenden Objekte werden mit Markern ausgestattet und von einem stationären Infrarotsensor getrackt. Die Polaris kann sowohl aktive als auch passive Marker tracken. Aktive Marker werden durch ein elektrisches Signal aktiviert und senden Infrarotstrahlung aus. Passive Marker besitzen mindestens drei reflektive Kugeln. Der Infrarotsensor der Polaris sendet Infrarotstrahlung aus, welche von den passiven Markern reflektiert wird. Anhand der erhaltenen Informationen kann der Infrarotsensor die Position und Rotation der Marker bestimmen.

**Technische Daten**

| Komponente | Daten |
|---|---|
| **Marker** | passiv, kabellos aktiv |
| **Max. Anzahl passiver Marker** | 25 passive Marker |
| **Max. Update Rate** | 60Hz (standard) / 250Hz (optional) |
| **Infrarotsensor Maße** | 591 mm x 103 mm x 106 mm |
| **Infrarotsensor Anbringung** | Anbringung erhöht an Wand oder Stativ<br>Fixierung an Rückseite |
| **Max. Messvolumen** | Beginn Messung in 950 mm Entfernung von Infrarotsensor<br>1856 mm x 1470 mm x 2050 mm (Breite x Höhe x Tiefe) |

## 4.3 Unity

*Unity* ist eine Gaming Engine entwickelt von Unity Technologies. Sie kann zum Entwickeln von 3D- und 2D-Spielen genutzt werden, sowie unter anderem zum Erstellen von AR- und VR-Anwendungen. Unity unterstützt mehr als 25 Plattformen, darunter auch die Universal Windows Plattform, welche auf der Hololens genutzt wird. Microsoft empfiehlt Unity zum Entwickeln von MR-Anwendungen für die Hololens, mit der Begründung, dass Unity 'die schnellste Methode zum Erstellen einer Mixed Reality-App' sei [16]. In dieser Arbeit wurde Unity 2018.2.21f1 genutzt.

## 4.4 Mixed Reality Toolkit

Das *Mixed Reality Toolkit* (MRTK) für Unity ist ein von Microsoft geleitetes Open Source Projekt zur Unterstützung und Beschleunigung der Entwicklung für die Hololens, Windows Mixed Reality immersive (VR) Headsets und OpenVR. Es enthält verschiedene Bauteile in Form von Skripten und vorgefertigten Objekten (Assets). Eines dieser Assets ist das *Input Manager Prefab*, das sich unter anderem aus einem *Gaze Manager*, einem *Input Manager* und einem *Focus Manager* Skript zusammensetzt. Mit Hilfe dieser Skripte lässt sich der Blick des Nutzers verfolgen und Gesten erkennen. Die Verfolgung der Blickrichtung ermöglicht das Anzeigen eines Cursors für den Nutzer und dadurch das Anfokussieren einzelner Objekte in der Szene. So besteht die Möglichkeit, Objekte bei Fokussierung farblich hervorzuheben oder weitere Informationen anzuzeigen, die sonst nicht sichtbar sind. In dieser Arbeit wurde die MRTK Version 2017.4.3.0 verwendet.

## 4.5 Vuforia Augmented Reality SDK

Vuforia ist ein plattformübergreifendes Software Development Kit zur Entwicklung von Mixed Reality Anwendungen. Vuforia lässt sich sowohl mit iOS- und Android-Geräten als auch mit der Hololens nutzen. Es ist in Unity integriert und ermöglicht durch Methoden der Computer Vision das Tracken von Markern in verschiedenen Formen, wie z.B. das Tracken eines einzelnen Bildes, 3D-Konfigurationen mit mehreren Bildern oder eines 3D Objektes. Vuforia ist nicht open source. Die in dieser Arbeit genutzte Version der Vuforia Engine ist 8.0.10.

Vuforia stellt in der Unity Szene ein eigenes Kamera-Asset zur Verfügung, welches in dieser Arbeit Anstelle des MixedRealityCameraParent-Assets aus dem Mixed Reality Toolkit verwendet wird.

### 4.5.1 Vuforia Model Targets

*Vuforia Model Targets* ermöglichen es, Objekte aus der realen Welt basierend auf ihrer Form zu Tracken. Dabei kann es sich um ein kleines Objekt wie ein Spielzeugauto handeln, aber auch um ein großes Objekt, wie ein richtiges Auto. Eine Voraussetzung hierfür ist jedoch, dass ein 3D-Modell (bspw. ein CAD-Modell) des Objektes existiert. Damit das Tracking eines Model Targets gut funktioniert, müssen sowohl das Objekt, als auch das CAD-Modell einige Anforderungen erfüllen.

Ein Objekt, das getrackt werden soll, sollte:

- Fix im Raum angebracht sein
- Eine farbige oder gemusterte Oberfläche besitzen
- Komplex sein
- Keine beweglichen Teile haben und nicht verformbar sein

Das CAD-Modell sollte:

- Keine Löcher oder Risse aufweisen
- Keine fehlenden Teile haben
- Keine Normalen aufweisen, die in eine andere Richtung als die Oberflächennormale zeigen
- Keine fehlenden Farbinformationen oder fehlende Textur aufweisen
- Keine falsche Textur aufweisen

#### Vuforia Model Target Generator

Um ein Model Target für die Verwendung in Unity zu erstellen, hat Vuforia den *Model Target Generator* (MTG) entwickelt. Mit Hilfe des MTGs wird aus einem CAD-Modell eine Vuforia Datenbank generiert, welche als *.unitypackage* heruntergeladen und in Unity importiert werden kann. Es ist möglich, die Datenbank mit Cloudbasiertem Deep-Learning zu trainieren und so eine *Advanced Model Target* Datenbank zu erhalten, dies wurde allerdings in dieser Arbeit nicht getan.

#### Guide Views

Bei einer untrainierten Datenbank muss das zu trackende Objekt von einer bestimmten Position und einem bestimmten Winkel aus mit der Hololens angeschaut werden, damit das Tracking beginnen kann. Dazu lassen sich im MTG ein oder mehr Bilder generieren, die während der Anwendung dem Nutzer angezeigt werden. Sie überlagern das Gesehene mit Kanten des 3D-Modells (siehe Abb.4.1). Der Nutzer muss sich dann so ausrichten, dass Objekt und Hilfsbild übereinstimmen. So hilft das Bild dem Nutzer so dabei, sich richtig zu positionieren und die Hololens im korrekten Winkel zum zu trackenden Objekt auszurichten. Als *Guide View* wird in Vuforia sowohl das Hilfsbild, als auch Winkel und Position, die relativ zum zu trackenden Objekt eingenommen werden müssen bezeichnet.

### 4.5.2 Vuforia Image Targets

Ein *Vuforia Image Target* ist eine Art Fiducial, das von Vuforia getrackt werden kann. Dabei muss zuerst eine Datenbank im Target Manager des Vuforia Developer Portals angelegt werden. Anschließend kann diese als *.unitypackage* heruntergeladen und in das Unityprojekt importiert werden. Die Vuforia Engine findet und trackt die natürlichen Merkmale in einem Target indem sie diese mit der Datenbank abgleicht.

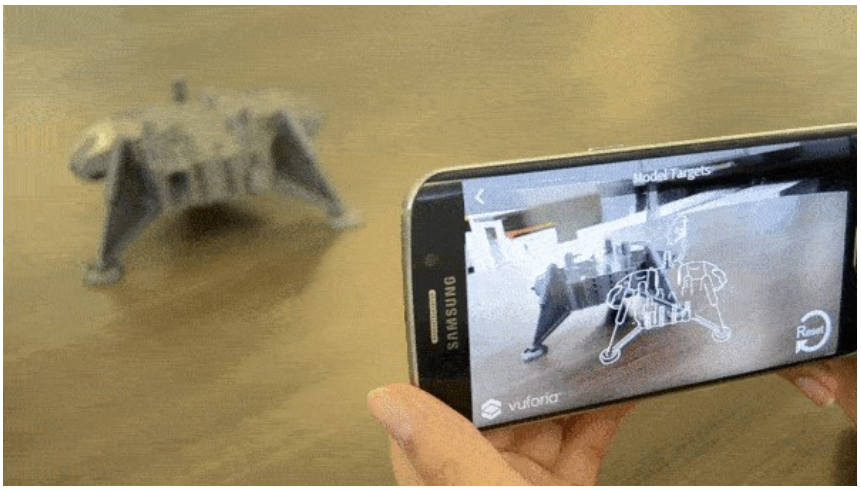


Abb. 4.1: Guide View auf einem Android Gerät (https://library.vuforia.com/features/objects/model-targets.html) 13.08.2019

In Unity kann einem Image Target ein GameObject als Kind hinzugefügt werden. Wird dieses Image Target erkannt und getrackt, wird das Kind des Image Targets gerendert. Vuforia trackt ein Target solange es teilweise im Sichtfeld der Kamera ist. Ist das Target nicht mehr im Sichtfeld der Kamera und kann somit nicht mehr getrackt werden, wird auch das Kind nicht mehr gerendert. Mit Hilfe von *Extended Tracking* kann die Robustheit des Trackings erhöht werden. Extended Tracking bedeutet, dass die Pose des Targets bekannt bleibt, obwohl das Target sich nicht mehr im Sichtfeld des Gerätes befindet. Dies wird mit Hilfe des *Positional Device Trackers* erreicht, der seit der Vuforia Engine 7.2 standardmäßig aktiviert ist. Durch Extended Tracking ist es möglich, komplexe Hologramme anzuzeigen, bei welchen das Target durch die Hologrammgröße bei Betrachtung nicht konstant im Sichtfeld ist oder bei welchen das Target durch Nutzung von Handgesten stellenweise verdeckt wird.

**Image Target Optimierung**

Das Vuforia Developer Portal bietet eine Bewertung von 0 bis 5 Sternen für hochgeladene Bilder an. Diese Bewertung sagt aus, wie gut sich das Bild als Image Target eignet, also wie gut das Bild mit Hilfe von Vuforia erkannt und getrackt werden kann. Je höher die Bewertung, desto besser eignet es sich. Ein Bild, welches als Image Target verwendet werden soll, sollte:

- detailliert sein,
- hohen Kontrast aufweisen und
- keine sich wiederholenden Muster haben.

Vuforia analysiert ein hochgeladenes Bild auf seine Merkmale und speichert Bild und Merkmale in einer Datenbank ab. Unter Merkmal versteht Vuforia ëin scharfes, spitzes, kantiges Detail im Bild". Desweiteren bezeichnet Vuforia die Merkmale als "kontrastbasierte Merkmale". Da Vuforia nicht open source ist, kann keine genauere Aussage über den von Vuforia genutzten Computer Vision Algorithmus zur Merkmalserkennung getroffen werden. Die gefundenen Merkmale eines Bildes werden im Target Manager des Vuforia Developer Portals als gelbe Kreuze angezeigt.

**Genutzte Image Targets**

Die in dieser Arbeit genutzten Marker wurden mit einem AR Marker Generator [1] erstellt.

[1] https://shawnlehner.github.io/ARMaker/ (Stand 06.07.2019)

Ihnen wurde eine Nummer in der linken oberen Ecke hinzugefügt, um sie unterscheidbar zu machen. Sie wurden auf nicht-glänzendem Papier in der Größe 5cm x 5cm gedruckt. Damit sie nicht flexibel sind, wurden sie auf einem Stück Pappe befestigt. Im Target Manager des Vuforia Developer Portals haben sie eine 5-Sterne-Bewertung erhalten, eignen sich also gut als Image Targets.

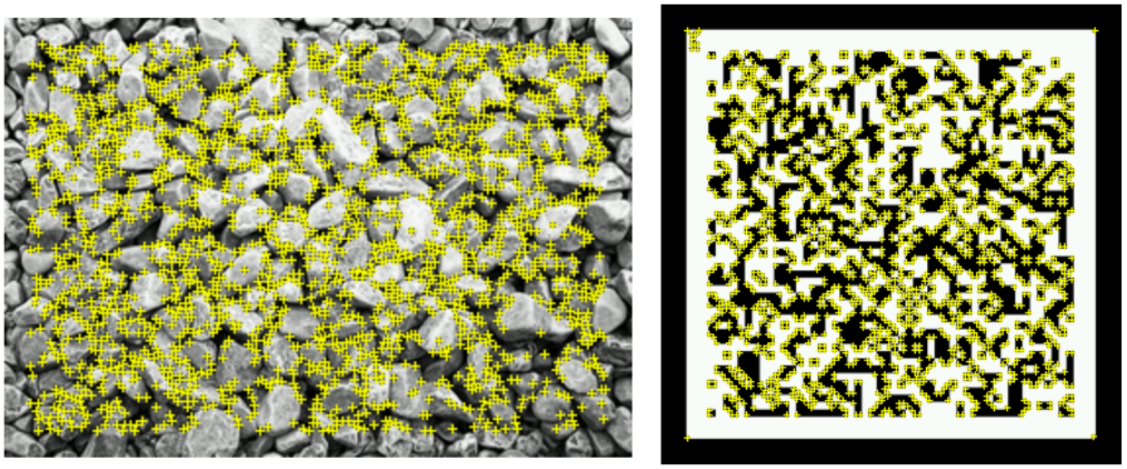

Abb. 4.2: Links: Merkmale im Beispielbild von Vuforia, Rechts: Merkmale der in dieser Arbeit verwendeten Marker

## 4.6 Versuchsaufbau

Ziel dieser Arbeit ist, ein Hologramm mit Hilfe der Hololens ohne externes Tracking System an einem Phantom zu Registrieren. Dazu muss das Phantom während der Laufzeit von der Hololens getrackt werden können. Um dies zu erreichen, wurden mehrere Möglichkeiten diskutiert. Einige dieser Möglichkeiten waren Vuforia Cylinder Targets, Point Fiducials und Aruco Marker.

Vuforia Cylinder Targets wurden ebenfalls von Frantz et al. für ihre Registrierung eines Schädels genutzt. Dabei handelt es sich um ein Bild, welches außen an einem Zylinder angebracht wird und getrackt werden kann auch wenn der Nutzer sich um den Zylinder herum bewegt. Diese Option wurde jedoch verworfen, da das in dieser Arbeit genutzte Phantom mit einem Torso wesentlich gößer als ein Schädel ist und es daher Probleme gegeben hätte, das Target immer im Blickfeld zu halten. Desweiteren hätte das Target nur schwierig stabil neben dem Phantom positioniert werden können.

Eine weitere Überlegung galt kreisförmigen einfarbigen Point Fiducials. In diesem Fall hätte auf den Kamerastream der Hololens zugegriffen werden müssen. In diesem hätten erst alle Pixel in der Farbe der Fiducials gefunden werden müssen und anschließend beispielsweise mit Hilfe der Hough-Transformation die Fiducials segmentiert werden müssen. Ein Vorteil bei kreisförmigen Fiducials ist, dass die Kreisform relativ invariant zu Verzerrungen ist. Allerdings käme es durch Unterschiede im Lichteinfall zu Farbvarianzen im Kamerastream, sodass es Probleme beim finden der korrekten Farbpixel gegeben hätte. Zusätzlich bietet ein kreisförmiger Fiducial keine 6DoF, sodass zu jeder Zeit mehrere Fiducials sichtbar sein müssten, um die Rotation der Kamera korrekt bestimmen zu können. Aus diesem Grund wurde auch diese Möglichkeit verworfen.

Bei der Nutzung von Aruco Markern bietet ein einzelner Marker durch seine planare quadratische Form 6DoF. Da mit Vuforia jedoch schon ein SDK existiert, welches ebenso Marker mit 6DoF bietet und schon in Unity integriert ist, wurden auch die Aruco Marker

verworfen. Implementiert wurde Vuforia Model Target Tracking und Vuforia Image Target Tracking, worauf im Folgenden genauer einegegangen wird.

### 4.6.1 Model Target Tracking am Phantom

Es wurde ein Vuforia Model Target Tracking implementiert. Getrackt wurde die Bauchplatte des Phantoms, da nur diese vollständig unbedeckt sichtbar ist. Das CAD-Modell der Bauchplatte wurde in den Model Target Generator geladen.

Ein Vorteile dieses Ansatzes ist, dass keine zusätzlichen Marker an den Patienten angebracht werden müssen, die etwas verdecken könnten. Allerdings ist dieser Ansatz dennoch nicht geeignet für eine Anwendung im medizinischen Bereich, da ein 3D-Modell eines realen Patienten nicht die Anforderungen erfüllt, die an ein Model Target gestellt werden. Zusätzlich ist der zeitige Aufwand, den das Erstellen eines Model Targets mit dem Model Target Generator mit sich bringt sehr viel höher als das Nutzen von planaren Markern, welche an den Patienten angebracht werden. Aus diesem Grund wurde im weiteren Verlauf auf Model Targets verzichtet.

### 4.6.2 Image Target Tracking

Die Registrierung von Hologramm und Objekt in der realen Welt wurde mit Hilfe von Vuforia Image Targets durchgeführt. Da das Phantom nicht durchgehend verfügbar war, wurde eine Vive Pro Box als zusätzliches Objekt genutzt, an welchem ein in Unity aus einem Würfel erstelltes 3D-Modell registriert wurde. Der Großteil der Entwicklung der Anwendung wurde an dieser Box durchgeführt. An Box und Phantom wurden Image Targets angebracht. Sie wurden so befestigt, dass sie mit der Hololens von möglichst vielen Positionen aus gut sichtbar sind. Beim Phantom wurden die Marker nicht auf der Bauchplatte angebracht, da diese abnehmbar ist und somit eine vorher durchgeführte Kalibrierung durch eine verschoben abgelegte Bauchplatte ungenau werden würde.

In Unity hängen die 3D-Modelle von Box und Phantom jeweils an einem leeren Gameobjekt. Dieses Gameobjekt besitzt ein Script *ModelPositionUpdater.cs* welches die Hologramme während der Laufzeit positioniert. Jedes Image Target hat ein *TargetData.cs* Script, in welchem Positions- und Rotationsoffset vom Target zum Ursprung des entsprechenden Modells (Box oder Phantom) nach Kalibrierung festgehalten wird. Jedes ImageTarget kennt also die Pose, an der das 3D-Modell in Relation zu sich selbst liegt. Wird ein Target während der Laufzeit erkannt und getrackt, nutzt der ModelPositionUpdater diesen Offset zu Positionierung des Hologramms. Aufgrund der 6DoF der Target wäre ein einzelnes Target theoretisch ausreichend, um das Hologramm korrekt zu positionieren. Da sich der Träger der Hololens jedoch um das Phantom bzw. die Box herum bewegen kann, ist ein Target nicht durchgehend sichtbar. Es kommt zusätzlich vor, dass aufgrund eines zu spitzen Winkels zwischen Target und Hololenskamera die Rotation des Targets von Vuforia nicht korrekt erkannt wird. Aus diesen Gründen wurden mehrere Targets angebracht. Der ModelPositionUpdater bildet aus den Offsetvektoren aller zurzeit getrackten und kalibrierten Targets, sowie auch aus den Offsetquaternionen dieser Targets das arithmetische Mittel. Ziel davon ist es, die Positionierung stabiler und genauer gegenüber eventuellen Abweichungen im Tracking oder falscher Kalibrierung eines Markers zu machen.

## 4.7 Kalibrierung

Damit das 3D-Modell, welches angezeigt werden soll, vom ModelPositionUpdater positioniert werden kann, müssen die Marker kalibriert werden, also Offsetrotation und Offsetposition der Image Targets in das TargetData Script geschrieben werden. Diese Kalibrierung kann auf zwei Wege vorgenommen werden: mit Hilfe der Hololens direkt in der Anwendung oder mit Hilfe eines externen Trackingsystems wie der Polaris.

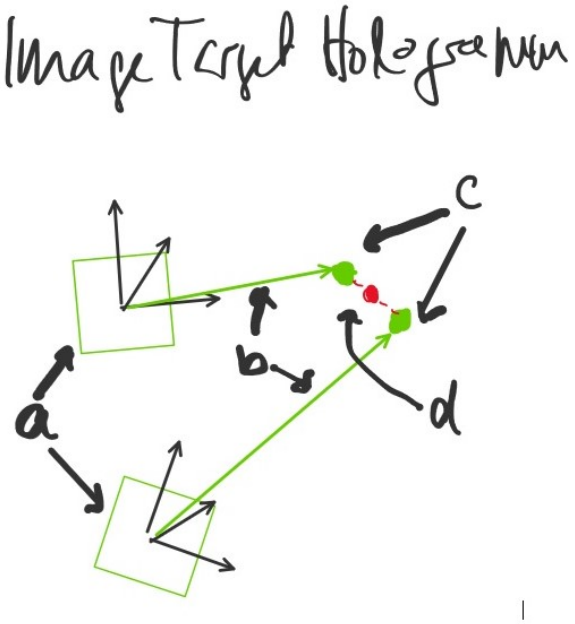


Abb. 4.3: **(a)** Marker mit lokalem Koordinatensystem; **(b)** Vektor von Marker zu (c); **(c)** Position, an welcher das 3D-Modell in Relation zum Marker laut TargetData Script liegt; **(d)** gemittelte Position und somit Position, an welcher das Hologramm dargestellt wird

### 4.7.1 Kalibrierung mit der Hololens

Zu Beginn der Kalibrierung mit der Hololens hängt das Hologramm an einem beweglichen Image Target. Mit Hilfe dieses Image Targets kann das Hologramm frei im Raum bewegt werden. Der Träger der Hololens nutzt diesen Marker nun dazu, das Hologramm so gut wie möglich auf dem Objekt (Vive Pro Box oder Phantom) zu positionieren. Der Marker wird auf dem Objekt abgelegt. Sobald einer der fest am Objekt angebrachten Marker im Sichtfeld der Hololens ist, kann die tatsächliche Kalibrierung per Knopfdruck gestartet werden. Das Hologramm hängt nun nicht mehr an dem beweglichen Marker, sondern an den Markern die zum Zeitpunkt des Knopfdrucks sichtbar waren. Nun ist es möglich, das Hologramm über ein Interface entlang X-, Y- oder Z-Achse zu verschieben oder um eine Achse zu rotieren. Ist der Nutzer zufrieden mit der Ausrichtung des Hologramms, wird ein *Save-Button* betätigt, welcher den Offset in Rotation und Position der zu diesem Zeitpunkt von der Kamera sichtbaren Marker in das jeweilige TargetData Script der Marker schreibt. Diese Marker gelten nun als kalibriert und werden von nun an zur Positionierung des Hologramms genutzt. Der Nutzer kann nun von einer anderen Position im Raum schauen, ob die Position des Hologramms korrekt ist. Ist sie dies nicht, kann weiterhin über das Interface eine Feinjustierung der Position und Rotation des Hologramms vorgenommen werden. Anschließend muss der Save-Button betätigt werden, um wiederum für alle zu diesem Zeitpunkt sichtbaren Marker den Offset in das TargetData Script zu schreiben und sie als kalibriert zu markieren. Ist ein bereits als kalibriert markierter Marker zu dem Zeitpunkt sichtbar, so wird der Offset in seinem TargetData Script überschrieben. Dieser Vorgang wird solange wiederholt, bis alle Marker als kalibriert markiert sind. Anschließend kann die Kalibrierung gespeichert und exportiert werden. Bei Neustart der Anwendung kann ebenso eine schon vorher abgeschlossene Kalibrierung geladen werden.

**Anpassung der Position über das Interface**

Zur Anpassung der Pose (Position und Rotation) über das Interface wurde ein temporärer Offset (*tempOffset*) eingeführt, welcher durch die Buttons des Interfaces geändert werden kann. Der Algorithmus zur Anpassung der Pose wurde in zwei Varianten implementiert.

**Variante 1**

1 Initiale Positionierung des Hologramms mit Hilfe des beweglichen Markers

2 Sobald mindestens ein anderer Marker sichtbar ist, Start des Kalibrierungsprozesses über Button *Start Calibration*, nutzen der zu diesem Zeitpunkt sichtbar gewesenen Marker zur Positionierung des Hologramms

3 Für als kalibriert markierte Marker gilt
$poseHologram = tempOffset \cdot markerOffset \cdot poseMarker$
wobei *markerOffset* der bereits kalibrierte Offset im TargetData Script ist

4 Anpassen von *tempOffset* über Interface Buttons

5 Speichern: für alle in diesem Moment sichtbaren Marker gilt
$markerOffset = tempOffset \cdot markerOffset$
Anschließend setzen von *tempOffset* auf *id* für alle Marker

Wie in Abb(noch einzufügen) zu sehen ist, kann es durch diese Methode zu großen Sprüngen des Hologramms kommen, wenn Marker im Sichtfeld sind, von denen einige einen *tempOffset* besitzen, bei anderen jedoch der $tempOffset = id$ ist. Aus diesem Grund wurde Variante 2 implementiert.

**Variante 2**

1 Initiale Positionierung des Hologramms mit Hilfe des beweglichen Markers

2 Sobald mindestens ein anderer Marker sichtbar ist, Start des Kalibrierungsprozesses über Button *Start Calibration*, nutzen der zu diesem Zeitpunkt sichtbar gewesenen Marker zur Positionierung des Hologramms

3 Für als kalibriert markierte Marker gilt
$poseEstimationHologram = markerOffset \cdot poseMarker$
$poseHologramFinal = tempOffset \cdot poseEstimationHologram$

4 Speichern: für alle in diesem Moment sichtbaren Marker wird der Offset von Hologramm zu Marker abgespeichert und $tempOffset = id$ gesetzt

### 4.7.2 Kalibrierung mit der Polaris

## 4.8 GUI und Interaktion

man muss mit dem Hologramm interagieren wegen kalibrierung. zwei möglichkeiten für GUI . 1. box transparent. 2. rendering reihenfolge (box muss von achsen überlagert werden).

für 2. entschieden, da bei box transparent buttons drin liegen und so nicht ersichtlich dass interagiert werden kann.

# 5. Evaluation

Description of your evaluation and its results. Interpretation of these results should go in the next chapter.

# 6. Diskussion

Discussion/Conclusion section. Some questions to think about while writing this section:

- What is the outcome of your evaluation?
- Did it meet the required goals defined in chapter 1? Why/Why not?
- How can future work improve on your work? Bessere Targets Bessere Positionierung Algorithmus, der Erkennt, wenn Target falsch getrackt wird (bspw dadurch, dass der winkel in dem es erkannt wird plötzlich stark anders ist)

# Abbildungsverzeichnis

# List of Algorithms

# Literaturverzeichnis


[1] H.-G. Ha and J. Hong, "Augmented Reality in Medicine," *Hanyang Medical Reviews*, vol. 36, no. 4, pp. 242–247, Nov. 2016. [Online]. Available: https://doi.org/10.7599/hmr.2016.36.4.242

[2] R. T. Azuma, "A Survey of Augmented Reality," *Presence: Teleoperators and Virtual Environments*, vol. 6, no. 4, pp. 355–385, Aug. 1997. [Online]. Available: https://doi.org/10.1162/pres.1997.6.4.355

[3] W. Birkfellner, M. Figl, K. Huber, F. Watzinger, F. Wanschitz, J. Hummel, R. Hanel, W. Greimel, P. Homolka, R. Ewers, and H. Bergmann, "A head-mounted operating binocular for augmented reality visualization in medicine - design and initial evaluation," *IEEE Transactions on Medical Imaging*, vol. 21, no. 8, pp. 991–997, Aug. 2002.

[4] H. Kenngott, A. Amin Preukschas, M. Wagner, F. Nickel, M. Müller, N. Bellemann, C. Stock, M. Fangerau, B. Radeleff, H.-U. Kauczor, H.-P. Meinzer, L. Maier-Hein, and B. Peter Müller-Stich, *Mobile, real-time, and point-of-care augmented reality is robust, accurate, and feasible: a prospective pilot study*, Jun. 2018, vol. 32.

[5] T. Frantz, B. Jansen, J. Duerinck, and J. Vandemeulebroucke, "Augmenting Microsoft's HoloLens with vuforia tracking for neuronavigation," *Healthcare Technology Letters*, vol. 5, no. 5, pp. 221–225, 2018.

[6] P. Pratt, M. Ives, G. Lawton, J. Simmons, N. Radev, L. Spyropoulou, and D. Amiras, "Through the HoloLens$^{TM}$ looking glass: augmented reality for extremity reconstruction surgery using 3d vascular models with perforating vessels," *European Radiology Experimental*, vol. 2, no. 1, p. 2, Jan. 2018. [Online]. Available: https://doi.org/10.1186/s41747-017-0033-2

[7] F. Liebmann, S. Roner, M. von Atzigen, D. Scaramuzza, R. Sutter, J. Snedeker, M. Farshad, and P. Fürnstahl, "Pedicle screw navigation using surface digitization on the Microsoft HoloLens," *International Journal of Computer Assisted Radiology and Surgery*, vol. 14, no. 7, pp. 1157–1165, Jul. 2019. [Online]. Available: http://link.springer.com/10.1007/s11548-019-01973-7

[8] E. Rae, A. Lasso, M. S. Holden, E. Morin, R. Levy, and G. Fichtinger, "Neurosurgical burr hole placement using the Microsoft HoloLens," in *Medical Imaging 2018: Image-Guided Procedures, Robotic Interventions, and Modeling*, vol. 10576. International Society for Optics and Photonics, Mar. 2018, p. 105760T. [Online]. Available: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/10576/105760T/Neurosurgical-burr-hole-placement-using-the-Microsoft-HoloLens/10.1117/12.2293680.short

[9] P. MILGRAM and F. KISHINO, "A Taxonomy of Mixed Reality Visual Displays," Dec. 1994. [Online]. Available: https://search.ieice.org/bin/summary.php?id=e77-d_12_1321

[10] B. Bray, v stdemo, J. McCulloch, N. Schonning, and M. Zeller, "What is mixed reality? - Mixed Reality." [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/mixed-reality

[11] M. Billinghurst, A. Clark, and G. Lee, "A Survey of Augmented Reality," *Foundations and Trends® in Human–Computer Interaction*, vol. 8, no. 2-3, pp. 73–272, Mar. 2015. [Online]. Available: https://www.nowpublishers.com/article/Details/HCI-049

[12] I. Rabbi and S. Ullah, "A Survey on Augmented Reality Challenges and Tracking Authors," p. 18, 2013.

[13] R. Szeliski, "Computer Vision: Algorithms and Applications," p. 979.

[14] "OpenHELP (Heidelberg laparoscopy phantom): development of an open-source surgical evaluation and training tool | SpringerLink." [Online]. Available: https://link.springer.com/article/10.1007/s00464-015-4094-0

[15] L. Qian, A. Barthel, A. Johnson, G. Osgood, P. Kazanzides, N. Navab, and B. Fuerst, "Comparison of optical see-through head-mounted displays for surgical interventions with object-anchored 2d-display," *International Journal of Computer Assisted Radiology and Surgery*, vol. 12, no. 6, pp. 901–910, Jun. 2017. [Online]. Available: https://doi.org/10.1007/s11548-017-1564-y

[16] Varnauld, "Mixed Reality documentation - Mixed Reality." [Online]. Available: https://docs.microsoft.com/en-us/windows/mixed-reality/