Hochschule Harz

Fachbereich Automatisierung und Informatik

Studiengang Medieninformatik

Bachelorarbeit zum Thema:

Innenraumausmessung und automatisierte Grundrisserstellung mithilfe der Microsoft HoloLens 2

Felix Altenhofen, m25563

Abgabe: 08.03.2021

Erstbetreuer: Prof. Jürgen Singer Ph.D.

Zweitbetreuer: Dipl.-Ing. Matthias Jantowski

I Inhaltsverzeichnis

II Abkürzungsverzeichnis

CAD..... Computer-aided Design (Computergestütztes Entwerfen)

HMD..... Head-mounted Display (Am Kopf befestigtes Anzeigegerät)

MRTK .....Mixed Reality Toolkit

WDP ..... Windows Device Portal

III Glossar

Anwendungsschnittstelle System einer Software, welches die Interaktion mit anderen Programmen ermöglicht beziehungsweise deren konkreten Ablauf definiert und somit die Anbindung der Software an diese gestattet.

Bildfrequenz Anzahl an unterschiedlichen Bildern, welche in einer bestimmten Zeitspanne auf einem Ausgabegerät, beispielsweise einem Bildschirm, dargestellt werden.

Computer-aided Design Entwurfs-, Konstruktions- und technische Zeichenaufgaben, welche computergestützt ausgeführt werden. [1]

Debugging Feststellen und Behandeln von Fehlern in Hard- oder Softwaresystemen.

Erweiterte Realität Form der Mixed Reality, bei der graphische digitale Inhalte über der realen Welt beziehungsweise einem Abbild dieser angezeigt werden, um sie zu erweitern. [2] Auch als „Augmented Reality“ bezeichnet.

Eye-Tracking Erfassung der Augenbewegungen und/oder Blickrichtung einer Person, häufig genutzt zur Interaktion mit digitalen Anwendungen.

Framework Für die Programmentwicklung genutztes Grundgerüst, welches häufig benötigte Funktionen beinhaltet, sodass diese von auf dem Framework basierenden Anwendungen genutzt werden können, ohne dort neu umgesetzt werden zu müssen.

Head-mounted Display Am Kopf getragenes Gerät, welches Bilder darstellt, üblicherweise auf einem vor den Augen platzierten Bildschirm.

Konsole Textbasiertes Ausgabefenster, welches bei Ausführung der zugehörigen Anwendung genutzt wird, um Informationen über ihren momentanen Zustand zu vermitteln.

Mixed Reality Systeme, in welchen die physische Welt beziehungsweise die menschliche Wahrnehmung dieser und digitale Inhalte in unterschiedlichem Ausmaß miteinander kombiniert werden. [2] Auch als „Gemischte Realität“ bezeichnet.

Polygonnetz Durch die Verbindung von Punkten geschaffene, aus Polygonen bestehende Repräsentation der Oberfläche eines oder mehrerer, meist dreidimensionaler Objekte.

Programmbibliothek	Sammlung an bezüglich ihrer Verwendung zusammengehörigen, vorgefertigten Softwarefunktionen, welche in unterschiedlichen Programmen verwendet werden können.
Punktwolke	Menge an Koordinatenpunkten im Raum, welche zusammen üblicherweise die Oberfläche eines dreidimensionalen Objektes abbilden.
Rastergrafik	Aus einer rasterförmig angeordneten Menge je eine Farbe darstellender Quadrate, sogenannter Pixel, bestehendes Bild. [3] Auch als Pixelgrafik bezeichnet.
Sprachsteuerung	System zur Interaktion mit einer Anwendung, welches auf gesprochene Anweisungen reagiert und ihnen entsprechende Aktionen ausführt.
Vektorgrafik	Aus, von Parametern definierten, geometrischen Formen bestehende Grafik, deren Inhalt rein mathematisch beschrieben wird und so unabhängig der Auflösung präzise dargestellt werden kann. [3]

IV Abbildungsverzeichnis

1 Einleitung

Infolge des rapiden technischen Fortschritts und seiner Auswirkungen auf die Wohnansprüche moderner Haushalte, stellt die Modernisierung von Gebäuden gerade in der heutigen Zeit einen gesellschaftlich enorm wichtigen Prozess dar. Infrastrukturelle Anpassungen und mit ihnen verbundene Bauvorhaben erfordern zur reibungslosen Durchführung die Realität wiederspiegelnde Gebäudepläne mit akkuraten Maßangaben. [4, p. 23] Gerade der Grundrissplan, eine zweidimensionale Abbildung eines horizontalen Schnittes durch die jeweils betrachtete Etage eines Gebäudes, stellt dabei die zentrale Grundlage besagter Prozesse dar.

Entsprechende Pläne sind für Bestandsbauwerke in vielen Fällen inkorrekt oder gänzlich nicht vorhanden. Demzufolge müssen ihre Inhalte durch Innenraumvermessung überprüft oder sogar von Grund auf neu erstellt werden. [4, p. 28] Selbst moderne, teilautomatisierte Vermessungsverfahren sind dabei jedoch mit großen Zeit- sowie Kostenaufwand verbunden und setzen zur Anwendung eine Vielzahl an Kenntnissen voraus.

Eine Alternative zu diesen etablierten Prozessen lässt sich möglicherweise in einer immer weiter verbreiteten Technologie, der *erweiterten Realität*, finden. Diese umfasst Verfahren zur Einbindung digitaler Inhalte in die reale Welt und wird gerade zur Unterstützung manueller Arbeitsabläufe genutzt. Geräte zur Darstellung von Inhalten der erweiterten Realität werden daher oftmals für die Verwendung in entsprechenden Arbeitsprozessen konzipiert.

Ein Vorreitergerät auf diesem Gebiet stellt die Microsoft HoloLens 2 dar, ein auf dem Kopf getragener Apparat, welcher digitale Inhalte direkt im Sichtfeld anwendender Personen einblendet. Um eine realistische Darstellung dieser digitalen Bilder zu ermöglichen, benötigt die HoloLens 2 dabei ein Verständnis ihres Umfeldes und erfasst zu diesem Zweck Umgebungsdaten. Dieser Prozess könnte dabei als intuitive Grundlage für ein Verfahren zur Innenraumvermessung genutzt werden.

Im Rahmen dieser Arbeit untersuche ich, ob es möglich ist, die HoloLens 2 zur Erfassung von Umgebungsdaten, mit anschließender Auswertung dieser zur Erstellung eines Grundrissplans, zu nutzen. Die Auswertung soll dabei automatisiert erfolgen, also möglichst wenige manuelle Aktionen durch die sie ausführende Person benötigen. Ich erörtere daraus hervorgehend, inwiefern ein solcher Prozess umsetzbar ist und als Alternative zu etablierten Verfahren gesehen werden kann.

Diese Überprüfung geschieht dabei praxisnah in einem Projekt zur beispielhaften Umsetzung eines solchen Prozesses, um ein für eine tatsächliche Anwendung des Verfahrens möglichst aussagekräftiges Testresultat zu erhalten. Es soll dabei eine konkrete Definition eines für den

erläuterten Zweck geeigneten Prozessablaufs erstellt und ein oder mehrere Programme zur Durchführung dessen entwickelt werden. Die dadurch erzielten Ergebnisse sollen daraufhin ausgewertet werden, um die Qualität des ermittelten Prozesses beziehungsweise seine Eignung für den beschriebenen Anwendungsfall beurteilen zu können.

Im Rahmen dieser Arbeit werde ich zunächst einen Überblick über aktuell etablierte Verfahren zur Innenraumvermessung geben und darauffolgend die Microsoft HoloLens 2 näher vorstellen. Als Nächstes folgt eine Darlegung des Projektaufbaus zur Auswertung der Realisierbarkeit eines Prozesses für den beschriebenen Anwendungszweck unter Verwendung der HoloLens 2. Dabei werden die notwendigen Einzelschritte definiert und Bewertungskriterien für die Auswertung des erzielten Ergebnisses festgelegt.

Im Anschluss erläutere ich die tatsächliche Projektdurchführung beziehungsweise den Ablauf der Umsetzung ihrer Einzelschritte und führe anschließend eine Bewertung des erreichten Resultats durch. Den Abschluss der Arbeit bildet ein Fazit zur durchgeführten Untersuchung sowie ein Überblick über das Einsatzpotential des ermittelten Prozesses und Möglichkeiten zur Weiterentwicklung des erzielten Standes beziehungsweise der zugrundeliegenden Technik.

2 Etablierte Vermessungsverfahren

Für die Vermessung von Objekten existieren diverse Verfahren, neben der klassischen Handvermessung auch verschiedene Methoden zur Vereinfachung beziehungsweise Automatisierung des Prozesses. Im Folgenden werde ich einen kurzen Überblick über die Grundlagen ausgewählter Verfahren geben, welche bei der Bauwerksvermessung hauptsächlich Anwendung finden.

Da die meisten der behandelten Messmethoden optische Verfahren zur Messung nutzen, wirkt sich bei ihnen die Oberflächenbeschaffenheit betrachteter Objekte stark auf die Richtigkeit der erzielten Ergebnisse aus. Auch andere Faktoren, beispielsweise die den Messbereich umschließende Atmosphäre, können einen Einfluss auf das Ergebnis haben. [5, p. 112] Dieser ist für die bei der Innenraumvermessung vorliegenden kurzen Messstrecken allerdings nicht signifikant.

Die vorgestellten Verfahren führen grundsätzlich zunächst eine Gesamterfassung der betrachteten Gebäudestrukturen durch, wobei eine Zusammensetzung verschiedener Messergebnisse notwendig ist, um das betrachtete Objekt in seiner Gesamtheit abzubilden. Aus dem gewonnenen Ergebnis werden daraufhin in einem sekundären Schritt Pläne, beispielsweise Grund- oder Aufrisse, erstellt. [4, p. 27] Dieser Prozess kann dabei komplett von Hand oder teilautomatisiert erfolgen. [4, p. 45]

Da für verschiedene Gebäude oder Gebäudeteile unterschiedliche Messverfahren zu einem in Bezug auf Aufwand und Genauigkeit optimalen Ergebnis führen können, ist auch eine Kombination der behandelten Messtechniken möglich.

2.1 Elektronisch unterstützte Handvermessung

Die Handermessung stellt die schlichteste und gerade im Amateurbereich verbreitetste Vermessungsmethode dar. Während sie ursprünglich komplett manuell mithilfe von Gliedermaßstab, Lot und ähnlichen Werkzeugen erfolgte, wird sie in der heutigen Zeit zumeist durch komplexere technische Geräte unterstützt. [4, p. 30] Üblicherweise werden handgeführte elektrooptische Distanzmessgeräte und/oder stationäre Tachymeter verwendet, die Messungen mithilfe der Laufzeitmessung oder des Phasenvergleichsverfahrens, welche ich in den folgenden Abschnitten näher erläutere, vornehmen. [4, pp. 30, 33]

Die die Messungen durchführenden Personen müssen sichergehen, dass alle benötigten Maße erfasst werden, um eine reibungslose Weiterarbeit zu ermöglichen. Die ermittelten Daten

können dabei direkt an einen Computer weitergeleitet und mithilfe geeigneter Software für die Erstellung von Plänen und/oder Modellen genutzt werden. [4, pp. 30-31] Das Zusammenfügen der ermittelten Messergebnisse zu einem Plan ist unabhängig davon üblicherweise mit hohem Aufwand verbunden. [4, pp. 31-32]

Die Handvermessung stellt für einfache Innenräume, welche wenige Einzelmessungen benötigen, zumeist die effektivste und kostengünstigste Vermessungsmöglichkeit dar, ist bei komplizierten geometrischen Strukturen aber sehr aufwändig und gerade in Verbindung mit Rundungen oft nicht genau durchführbar. [4, p. 43] Zudem kann die Genauigkeit ihres Ergebnisses bei vielen Teilmessungen stark reduziert sein, da kein übergeordnetes Referenzsystem existiert, wodurch kleine Ungenauigkeiten, welche gerade bei Handgeräten auftauchen, sich bei verschiedenen genommenen Maßen schnell zu großen Fehlern addieren können. [4, pp. 32-33]

2.2 Laserscanning

Das Laserscanning ist ein sehr verbreitetes Verfahren zur Objektvermessung. Dazu wird ein Laser genutzt, welcher die Umgebung mit vielen aufeinanderfolgenden Punktmessungen in einem Muster abtastet. Für jeden Messpunkt wird dabei die Entfernung und damit seine Position relativ zum Laserursprung bestimmt, wodurch eine dreidimensionale *Punktwolke* erzeugt werden kann.

Das einfachste zur Distanzbestimmung eingesetzte Verfahren ist die Laufzeitmessung. [5, p. 109] Sie basiert darauf, dass eine Schall- oder Lichtwelle an einem Ende der zu messenden Strecke abgegeben und am anderen Ende dieser reflektiert wird. Die Zeit bis zur Rückkehr wird dabei gemessen und anhand der bekannten Geschwindigkeit der verwendeten Welle der zurückgelegte Weg ermittelt.

Neben dem Laufzeitverfahren wird für Laserscanning alternativ das Phasenvergleichsverfahren zur Distanzbestimmung verwendet, wobei die Abtastung des zu vermessenden Objekts unverändert bleibt. Diese Methode basiert darauf, dass ein Laser eine Welle ist und dadurch eine periodisch wiederholte Form hat. So kann durch Vergleich der Positionen von Wellenbergen beziehungsweise Wellentälern bei Abgabe und späterer Rückkehr des Lasers die Länge der vom ihm zurückgelegten Strecke ermittelt werden. [5, p. 119]

Ein identischer Positionsunterschied kann dabei bei verschiedenen Entfernungen entstehen, die möglichen Distanzen sind aber abhängig von der Frequenz des genutzten Lasers. [5, p. 120] Dadurch ist es möglich, den Phasenunterschied einer konkreten Entfernung zuzuordnen, indem

bei jedem Messpunkt mehrere Laser mit unterschiedlichen Frequenzen zur Messung verwendet werden. [5, p. 121] Die korrekte Distanz ist dann die Einzige allen Lasern als mögliche Entfernung Gemeinsame.

Das Laufzeitmessverfahren bietet dabei, bei bezüglich ihrer Kosten vergleichbaren Geräten, eine geringere Genauigkeit als die phasenbasierte Messung. [5, pp. 121-122] Aus diesem Grund ist das Phasenvergleichsverfahren gerade bei den für die Innenraumvermessung relevanten kurzen Distanzen für akkuratere Ergebnisse vorzuziehen. [5, pp. 109-110]

Fehlerhafte Messungen, welche beispielsweise durch Mehrfachreflektion entstehen können, sind gerade bei einer hohen gewählten Messpunktdichte gut feststellbar und leicht zu entfernen. [4, p. 44] Da die einzelnen Messungen jedoch aufeinanderfolgen, ist die Datenerfassung mit einem hohen Zeitaufwand verbunden, welcher sich bei einer, für ein feineres Ergebnis nötigen, höheren Dichte an Messpunkten entsprechend weiter steigert.

Laserscanning bietet auch die Möglichkeit, die Intensität des zurückkehrenden Lasers zusätzlich zur gemessenen Entfernung zu erfassen. Dadurch kann ein, die Reflektivität der jeweils für die Messung betrachteten Oberfläche darstellendes, Graustufenbild erzeugt werden. Dieses kann bei der Einordnung und/oder Bereinigung der erfassten Daten helfen.

Auf dem beim Laserscanning verwendete Prinzip der Laufzeitmessung basiert auch das sogenannte „Lidar“, welches zur Umgebungserfassung beispielsweise bei Fahrzeugen genutzt wird. [6] Statt aufeinanderfolgender Einzelmessungen können zudem viele räumlich verteilte Messungen auf einmal durchgeführt werden, um Strukturen flächendeckend zu erfassen. Dieses Konzept findet bei Time-of-Flight-Sensoren Anwendung, welche unter anderem in Mobilgeräten genutzt werden. Dort dienen sie beispielsweise der Verarbeitung aufgenommener Fotografien, für welche die Abstände verschiedener Bildbestandteile zum Gerät ausgewertet werden. Die Erfassung der jeweiligen Entfernungen geschieht dabei durch großflächig abgegebene, für Menschen unsichtbare Infrarotlichtwellen. Jedes dieser beiden Systeme kann potentiell auch zur Vermessung verwendet werden.

2.3 Photogrammetrie

Die Photogrammetrie ist ein Verfahren, in welchem aus Fotografien eines oder mehrerer Objekte deren Maße, Lage und Form bestimmt werden. Dabei wird aus den erstellten Bildern mithilfe einer Software automatisiert, unter Betrachtung der Position und Ausrichtung der Kamera zum Zeitpunkt der Aufnahme sowie ihren physikalischen Gegebenheiten, beispielsweise durch ihre Linse hervorgerufene Verzerrungen, die Oberflächenform der

betrachteten Geometrien ermittelt. [7] Neben speziell zu diesem Zweck entwickelten Messkameras können für den Prozess auch gewöhnliche Kameras eingesetzt werden. [7]

Bereits ein einzelnes Bild kann dabei unter Umständen zur Ableitung eines zweidimensionalen Modells des betrachteten Objekts verwendet werden. [4, pp. 36-37] Für eine akkuratere, detaillierte Auswertung als dreidimensionales Modell ist hingegen eine große Menge einander überlappender Bilder des Objekts aus verschiedenen Betrachtungswinkeln notwendig. [4, p. 37] Die für ein realitätsnahes Ergebnis nötige Bildanzahl steigt dabei mit der Komplexität und Größe des betrachteten Objekts und zieht entsprechenden höheren Aufwand bei Erfassung und Auswertung nach sich.

Die Photogrammetrie setzt dabei gute Lichtverhältnisse voraus, in welchen aussagekräftige Bilder erzeugt werden können. Objektbereiche ohne deutliche Konturen sind mit dieser Methode deutlich schwerer oder gar nicht erfassbar und bedingen unter Umständen fortschrittlichere Varianten des Verfahrens, beispielsweise die Stereophotogrammmetrie, welche deutlich zeitintensiver sind. [4, p. 39] Aufgrund des generell hohen zeitlichen Aufwands, den die Photogrammetrie voraussetzt, ist sie für die Vermessung von kleinen und/oder geometrisch komplexen Innenräumen ungeeignet. [4, pp. 35-36] Es ist zudem zu beachten, dass das Ergebnis diese Verfahrens vollständig auf der Interpretation der erstellten Bilder basiert und nicht auf akkuraten Messungen.

3 Technik der Microsoft HoloLens 2

Bei der HoloLens 2 handelt es sich um ein 2019 von der Microsoft Corporation veröffentlichtes *Head-mounted Display* (HMD) zur Darstellung von *Mixed-Reality*-Inhalten. Sie zeigt dabei die tatsächliche Umgebung der sie tragenden Person und erweitert diese durch digital im Sichtfeld platzierte, von Microsoft als „Hologramme“ bezeichnete Objekte. [8, pp. 48-49] Die so erzeugten Inhalte fallen damit konkreter in den Bereich der erweiterten Realität. Das Gerät ist eine Weiterentwicklung der 2016 von Microsoft veröffentlichten HoloLens.

Über die grundlegenden funktionalen Aspekte der HoloLens 2 sowie speziell die für den in dieser Arbeit betrachteten Anwendungsfall relevanten Funktionen und Einschränkungen des Geräts verschaffe ich im Folgenden einen kurzen Überblick.

3.1 Grundlegende Funktionen

Die HoloLens 2 ist ein eigenständiger Computer mit einer angepassten Version von Microsofts Betriebssystem Windows 10 [8, p. 48] [9], unterstützt durch die Windows-Mixed-Reality-Plattform, welche die einheitliche, einfache Entwicklung von Mixed-Reality-Anwendungen ermöglichen soll. Diese ist dabei die Nachfolgeplattform des ursprünglich rein für die HoloLens ausgelegten Windows Holographic.

Das HMD verfügt über je einen lichtdurchlässigen Bildschirm pro Auge, welcher das Einblenden digitaler Inhalte über der realen Welt ermöglicht. Dabei können Objekte dreidimensional dargestellt und im umgebenden Raum fixiert platziert werden, sodass sie sich perspektivisch in die Umgebung einfügen und greifbar erscheinen. Alle auf dem Gerät präsentierten Inhalte, inklusive der gesamten Bedienoberfläche, werden dabei auf diese Art als dreidimensionale Objekte repräsentiert, wie in Abbildung 1 zu sehen. Zudem kann Audio wiedergegeben werden, wobei das Klangbild entsprechend der virtuellen Position der Audioquelle relativ zur Person, welche die HoloLens 2 verwendet, angepasst wird. [9]

Abbildung 1: Bedienoberfläche der Microsoft HoloLens 2 mit im dreidimensionalen Raum platzierten, Anwendungen abbildenden Paneelen.

Mit dargestellten Inhalten kann auf verschiedene Arten interagiert werden. So können Objekte anhand der Kopfausrichtung der das HMD tragenden Person oder mithilfe von *Eye-Tracking* über ihre konkrete Blickrichtung für Aktionen ausgewählt und Interaktionen mit ihnen durch per *Sprachsteuerung* erfasste Befehle ausgelöst werden. [8, pp. 50-55] [10] Auch können Handgesten für die Steuerung genutzt werden, da die Hand- sowie Fingerpositionen der anwendenden Person erfasst werden, wodurch Hologramme beispielsweise gegriffen und bewegt werden können, als wären sie reale Objekte im Raum. [8, pp. 54-55]

Des Weiteren ist das HMD, neben diversen Sensoren zur Interaktion mit seiner Umgebung, mit einer Kamera ausgestattet, welche Außenstehenden die Betrachtung und/oder Unterstützung damit ausgeführter Arbeiten ermöglichst. [9] Dadurch, dass es nicht auf einen externen Rechner angewiesen ist und über einen Akku mit einer Laufzeit von mehreren Stunden verfügt, kann das Gerät kabellos verwendet werden und ist entsprechend flexibel einsetzbar. [9]

3.2 Unterschiede zum Vorgängermodell

Im Vergleich mit dem Vorgängermodell HoloLens, verfügt die HoloLens 2 über ein mehr als doppelt so großes Sichtfeld, also des Bereichs, in welchem Hologramme dargestellt werden können. [10] [11] Sie beinhaltet zudem neben einem regulären Prozessor eine überarbeitete Version der in der ursprünglichen HoloLens integrierten „Holo-Processing-Unit“, ein von Microsoft speziell für diese Geräte entwickelter Prozessorchip zur Berechnung der dargestellten Hologramme. [11]

Die Möglichkeiten zur Interaktion mit den Hologrammen wurden ebenfalls überarbeitet, wobei Eye-Tracking als neue Eingabemethode hinzugefügt wurde. [10] [11] Auch wurden die zur Steuerung verwendeten Gesten angepasst, um intuitiver verständlich zu sein. [10]

Das Design des Geräts wurde angepasst, um seinen Tragekomfort zu verbessern und so eine längere, angenehme Verwendung zu ermöglichen. [10] [11] Zudem kann der vordere Teil des HMD, welcher die Bildschirme und Sensoren des Geräts beinhaltet, nach oben geklappt werden, um die erweiterte Realität der HoloLens 2 schnell aktivieren beziehungsweise deaktivieren zu können. [10]

3.3 Projektrelevante Funktionen

Für die korrekte Platzierung von Hologrammen sowie ihre Verdeckung durch und Interaktion mit Objekten der realen Umgebung, benötigt die HoloLens 2 ein Verständnis ihrer Umgebung. [12, pp. 26-27] Zu diesem Zweck existiert das System zur räumlichen Abbildung, welches mithilfe sogenannter Time-of-Flight-Sensoren, basierend auf dem Prinzip der Laufzeitmessung, die Umgebungsgeometrie erfasst und in ein *Polygonnetz* umwandelt. [12, p. 27] Das so erstellte dreidimensionale Modell soll dabei die tatsächliche geometrische Beschaffenheit der Umgebung möglichst genau abbilden, um Hologramme akkurat mit ihr interagieren zu lassen. [12, p. 27] Gleichzeitig wird es, bedingt durch die begrenzte Leistung der HoloLens 2, durch einen nicht zu hohen Detailgrad ressourcensparend angelegt, wodurch filigranere Objekte oder Geometrien unter Umständen nicht wahrgenommen werden, die für die Innenraumvermessung relevanten gröberen Umgebungsmerkmale jedoch gut erfasst werden sollten. [12, p. 28]

Das so erzeugte Polygonnetz kann dabei ausgegeben und somit weiterverarbeitet werden. Auch kann es durch die HoloLens 2 in Echtzeit dargestellt werden und so als Feedback bezüglich der Beschaffenheit der erfassten Geometrie dienen. [13] Dabei werden die Kanten des Netzes über den entsprechenden Umgebungsbereichen dargestellt, wie in Abbildung 2 zu sehen.

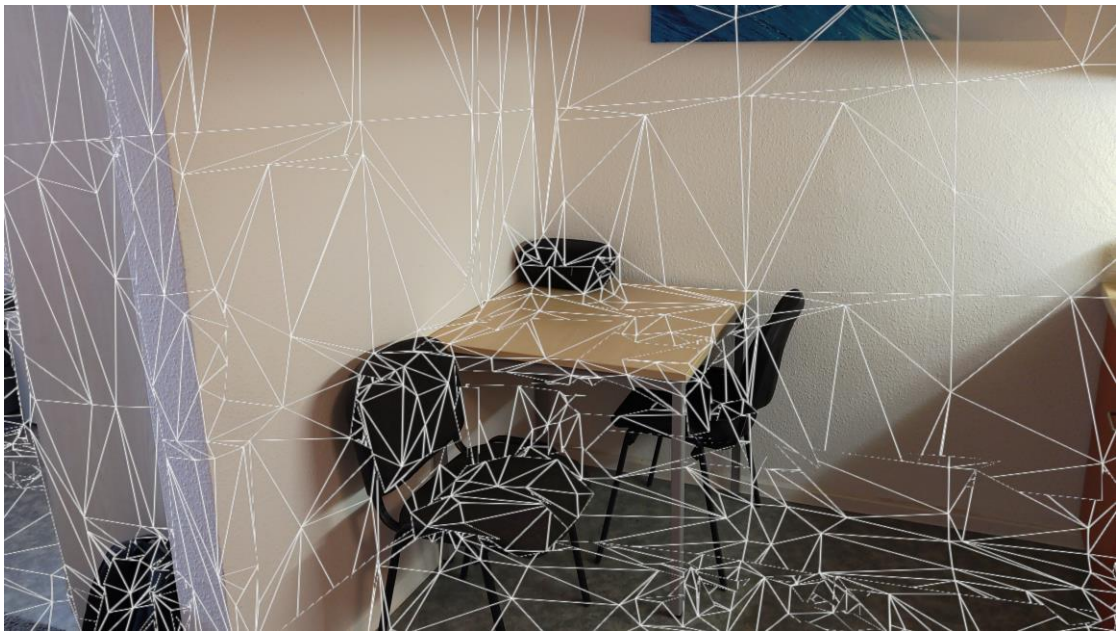

Abbildung 2: Über der Umgebung liegendes Polygonnetz des Systems zur räumlichen Abbildung.

Der Bereich, in welchem Umgebungsdaten gesammelt werden, ist auf einen Ausschnitt des Sichtfeldes der das HMD tragenden Person, in einer Reichweite von ein paar Metern, beschränkt. [12, pp. 27-28] Dabei werden bereits erfasste Umgebungen von der HoloLens 2 wiedererkannt und ihre bestehende Geometrie bearbeitet, statt sie komplett neu anzulegen. So können falsch oder nicht erfasste Teile des Polygonnetzes im Nachhinein korrigiert werden, ohne die gesamte Umgebung neu erfassen zu müssen.

Microsoft bietet zur Erweiterung dieser Funktionen zudem das Szenenverständnis-System an. Dieses kann die erfasste Umgebungsgeometrie teilweise interpretieren und so unvollständig erfasste Umgebungen komplettieren und/oder vereinfachen, wie in Abbildung 3 zu sehen. [14] Diese automatische Verarbeitung der wahrgenommenen Daten stellt dabei den üblichen Verwendungszweck des Systems dar, eine Ausgabe des zugrundeliegenden Polygonnetzes ist jedoch ebenfalls möglich und geschieht, wie in Abbildung 4 zu sehen, analog zur Darstellung im System zur räumlichen Abbildung. Im Gegensatz zum System zur räumlichen Abbildung werden dabei nicht konstant aktuelle Daten zur Verfügung gestellt, sondern nur auf manuelle Anfrage für die aktuellen Umgebungsinformationen ausgewertet, da die nötige Verarbeitung einen sehr rechenaufwändigen Prozess darstellt. [15]

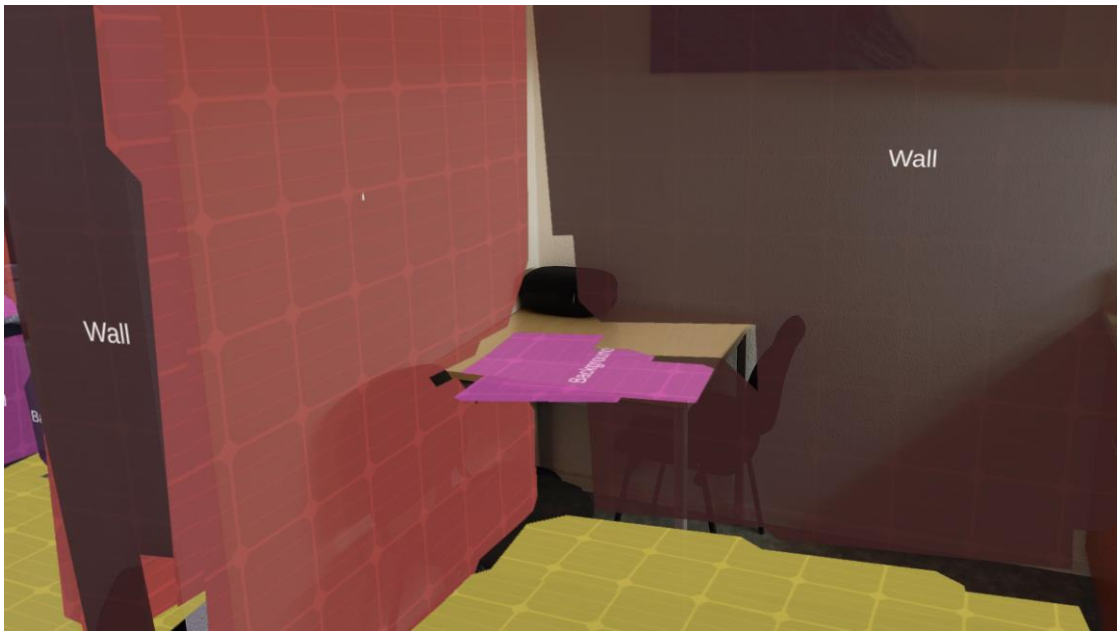

Abbildung 3: Über der Umgebung liegende, vom Szenenverständnis-System interpretierte Oberflächen.

Abbildung 4: Über der Umgebung liegendes Polygonnetz des Szenenverständnis-Systems.

Während das System zur räumlichen Abbildung auf die Betrachtung des Polygonnetzes im unmittelbaren Umfeld der die HoloLens 2 tragenden Person beschränkt ist, kann das Szenenverständnis-System zudem auf erfasste Umgebungsdaten in einem beliebigen Umkreis zugreifen. [14] Dadurch ist es für das zu bearbeitende Projekt potentiell von Relevanz, sofern die normalerweise begrenzte Zugriffsreichweite sich für die Vermessung größerer Räume beziehungsweise Raumkomplexe als problematisch erweisen könnte.

3.4 Für optimale Funktion zu beachtende Umgebungseigenschaften

Um eine optimale Funktionsweise der HoloLens 2 beziehungsweise ihrer Systeme zur Umgebungserfassung zu gewährleisten, muss die betrachtete Umgebung einige Voraussetzungen möglichst gut erfüllen. Während diese Umgebungseigenschaften und ihre Auswirkungen auf die erzeugten Ergebnisse in dieser Arbeit nicht tiefgehend analysiert werden sollen, sind sie dennoch zu beachten, um optimale Ergebnisse zu erzielen.

Grundsätzlich sollte die zu erfassende Umgebung, um ein optimales Funktionieren der Time-of-Flight-Sensoren zu gewährleisten, gleichmäßig, weder zu hell noch zu dunkel, beleuchtet sein. [16] Auch kann eine Umgebung mit vielen, möglichst einzigartigen Merkmalen, beispielsweise leicht identifizierbaren Objekten, von der HoloLens 2 einfacher erkannt und ihre Position bei Bewegung besser stabilisiert werden. [16] Generell sollten die betrachteten Räumlichkeiten möglichst gut unterscheidbar sein, um zu verhindern, dass ähnlich aussehende Räume irrtümlich als derselbe Raum wahrgenommen werden. [16]

Reflektionen oder transparente Objekte können zudem Erfassungsfehler, von Microsoft als „Halluzinationen“ bezeichnet, hervorrufen. [13] Dadurch entstehen im Polygonnetz Geometrien, welche in der Realität nicht existieren. Zudem können Objekte potentiell andere Objekte teilweise oder ganz verdecken, wodurch diese in der erfassten Geometrie fehlen. Diese Problematik tritt gleichermaßen bei anderen optischen Messverfahren auf, kann bei der HoloLens 2 durch ihre Bewegungsfreiheit jedoch besser ausgeglichen beziehungsweise korrigiert werden.

4 Projektaufbau

Vor Projektbeginn waren zunächst die konkreten Ziele des Projektes festzulegen und dessen Ablauf sowie das Vorgehen in den ermittelten Teilschritten zu planen. Auch mussten für die Entwicklung genutzte Programme, beziehungsweise *Anwendungsschnittstellen*, definiert werden. In den folgenden Abschnitten erläutere ich die Planung des Projektes beziehungsweise die generelle Vorbereitung seiner Bearbeitung.

4.1 Nutzung der HoloLens 2

Da die HoloLens 2 als mobiles Gerät nicht auf eine starke Rechenlast ausgelegt ist, war zunächst zu klären, ob alle im Projekt inbegriffenen Teilschritte direkt in einer Anwendung auf dem HMD geschehen sollten. Alternativ würde lediglich die Umgebungserfassung auf der HoloLens 2 ablaufen, während die Weiterverarbeitung der gesammelten Daten in eine separate Anwendung auf einen externen Rechner ausgelagert werden würde. Erstere Variante hätte dabei den Vorteil, dass der erzeugte Plan bereits während der Datenerfassung visualisiert werden und dadurch beispielsweise eine direkte Korrektur falsch oder unzureichend erkannter Umgebungselemente ermöglichen könnte.

Beim Testen einer einfachen Anwendung zum Erfassen und Visualisieren der momentanen Umgebung der HoloLens 2 stellte ich fest, dass das Gerät durch diese Funktionen bereits ausgelastet wurde. So wäre eine Erweiterung der Anwendung zur Weiterverarbeitung der Daten nur in Verbindung mit einem deutlich schlechteren Nutzungserlebnis durch eine niedrige und/oder unregelmäßige *Bildfrequenz* möglich gewesen.

Entsprechend entschied ich mich dafür, lediglich die Datenerfassung auf der HoloLens 2 auszuführen. Diese sollte dabei durch Hilfssysteme, beispielsweise die Visualisierung des die aktuelle Umgebung darstellenden Polygonnetzes, für Personen, welche die Erfassung ausführen, erleichtert werden.

4.2 Entwicklungsumgebung

Um den Aufwand zur Erstellung der für das Projekt benötigten Anwendungen für die HoloLens 2 und/oder externe Computer zu verringern, soll eine etablierte Entwicklungsumgebung Verwendung finden. Diese soll möglichst viele grundlegende Funktionen, welche in den Programmen zur Anwendung kommen, bereits implementieren, um den Entwicklungsaufwand

auf konkret für dieses Projekt relevante Aspekte zu beschränken. Diese Funktionen umfassen dabei beispielsweise die Kompatibilität mit den gängigen Betriebssystemen der verwendeten Geräte, die Darstellung grafischer Inhalte und Möglichkeiten zur Interaktion mit besagten Inhalten.

Das von Microsoft bereitgestellten Mixed-Reality-Toolkit (MRTK) ist eine *Programmbibliothek* zur Anwendungsentwicklung für Mixed-Reality-Systeme und stellt eine zentrale Komponente zur Entwicklung von Programmen für die HoloLens 2 dar. Es ersetzt die Bibliothek „HoloToolkit“, welche auf die ausschließliche Anwendungsentwicklung für die HoloLens ausgelegt war. Für die HoloLens 2 bestimmte Programme sollten in einem mit dem MRTK kompatiblen *Framework* unter Verwendung der MRTK-Bibliothek erstellt werden.

Ich legte mich dabei auf das Programm „Unity“ fest, wobei es sich um eine Spiel-Engine handelt, einer Entwicklungsumgebung, welche auf die einheitliche, schnelle Erstellung von Echtzeitanwendungen mit zwei- und/oder dreidimensionalen Inhalten ausgelegt ist und hauptsächlich der Entwicklung digitaler Spiele dient. Ich entschied mich dafür, da Unity einen hohen Funktionsumfang, inklusive Export der erstellten Anwendungen auf alle aktuell gängigen Betriebssysteme, bietet. Da für Unity-Anwendungen die verbreitete Programmiersprache C# genutzt wird, kann zudem für die Entwicklung auf eine Vielzahl durch Dritte erstellter Programmbibliotheken zur Funktionserweiterung zurückgegriffen werden.

Auch werden die Entwicklungswerkzeuge für HoloLens-Anwendungen von Microsoft bereits seit der Erstveröffentlichung der HoloLens für Unity zur Verfügung gestellt und sind entsprechend robust in die Entwicklungsumgebung integriert, wodurch Unity als eine primäres Entwicklungswerkzeug für die HoloLens 2 empfohlen wird. [8, p. 49] [17] In diesem Projekt soll Unity dabei nach Möglichkeit für alle nötigen Schritte, von der Datenerfassung bis zum Planexport, genutzt werden.

Unity-Anwendungen bestehen aus Szenen, dreidimensionalen Räumen, in welchen Objekte platziert werden. Alle diese Objekte beziehungsweise ihre Eigenschaften können mithilfe von Skripten, kleine, in der Programmiersprache C# geschriebene Programme, manipuliert werden. In den Szenen werden dabei auch Kameras platziert, die den Szeneninhalt aus ihrer Perspektive einfangen und als Ansicht für die die Anwendung nutzenden Personen ausgeben. Neben diesen grundlegenden Elementen, bietet Unity zudem weitere Komponenten zur Erweiterung dieser Basis, beispielsweise durch Audio oder Eingabeelemente.

Im Unity-Editor kann dabei neben der fertigen Anwendung, welche in der sogenannten Spielansicht dargestellt wird, auch mit der aktuellen Szene selbst interagiert werden. Dafür dient die Szenenansicht, welche die Szene aus Sicht einer in ihr frei beweglichen Kamera darstellt und es ermöglicht, alle darin befindlichen Objekte auszuwählen und direkt zu manipulieren.

Dabei werden alle änderbaren Eigenschaften des jeweils ausgewählten Objekts im sogenannten Inspektor aufgelistet und können dort entsprechend angepasst werden.

4.3 Erfassung und Export von Umgebungsdaten

Als Grundlage für den zu erzeugenden Grundriss muss zunächst auf die von der HoloLens 2 ermittelten Umgebungsdaten zugegriffen werden. Da ein externes Programm zur Weiterverarbeitung genutzt werden soll, müssen die erfassten Daten dabei zunächst von der HoloLens 2 exportiert werden, wobei ein möglichst universelles Format zum Einsatz kommen sollte. Dieses sollte die erhaltenen Daten zudem so repräsentieren, dass sie ohne starke Verarbeitung beziehungsweise Interpretation verwendet werden können.

Die Erfassung der aktuellen Umgebung geschieht durch das System zur räumlichen Abbildung dabei regelmäßig automatisch. Personen, welche das System nutzen, sollte dabei mithilfe einer auf dem HMD laufenden Anwendung Feedback zum aktuellen Stand der Umgebungserfassung gegeben werden. Aus den gewonnenen Daten wird zudem ein die Umgebung repräsentierendes Polygonnetzes erzeugt, auf welches von der aktuell laufenden Anwendung zugegriffen werden kann. Auch ein Export als dreidimensionales Modell in das gängige .obj-Format ist auf verschiedene Arten möglich.

Eine Möglichkeit dazu stellt das Windows Device Portal (WDP) dar, eine auf einem externen Rechner ausführbare Webanwendung, über die auf eine im gleichen lokalen Netzwerk befindliche HoloLens 2 zugegriffen werden kann. Über diese Anwendung kann das aktuelle Umgebungsmodell betrachtet und auf den externen Rechner übertragen werden. Diese Möglichkeit des Exports setzt voraus, dass beide involvierte Geräte mit einem Netzwerk verbunden sind, erschwert also unter Umständen die Anwendung des Verfahrens.

Alternativ kann der Datenexport auch über eine auf der HoloLens 2 ausgeführte Anwendung geschehen, wobei alternativ zum System für räumliche Abbildung auch Daten des Szenenverständnis-Systems genutzt werden können. Diese Variante sollte dabei mehr Kontrolle über das erzeugte Polygonnetz bieten, da beispielsweise ein höherer Detailgrad und der konkret zu betrachtende Umgebungsausschnitt festgelegt werden können. Die exportierten Daten können wahlweise auf der HoloLens 2 gespeichert und später über das WDP auf einen externen Rechner übertragen werden oder, bei Vorhandensein einer Internetverbindung, direkt nach dem Export auf einen Server hochgeladen werden.

Nach Möglichkeit sollen alle erwähnten Methoden getestet werden, um letztlich die am besten für unseren Anwendungsfall geeignete Variante zu ermitteln. Dabei soll das erzielte

Exportergebnis für den weiteren Projektablauf geeignet und der Arbeitsablauf zur Durchführung des Exports möglichst unkompliziert gestaltet sein.

4.4 Extrahierung eines Grundrisses aus den erfassten Daten

Im nächsten Schritt ist aus den exportierten Umgebungsdaten ein Grundriss zu erstellen. Dabei sollen das als Grundlage genutzte Polygonnetz sowie der daraus ermittelte Grundrissplan für die Personen, welche die Anwendung nutzen, dargestellt werden und auf visueller Basis Anpassungen ermöglichen. Nach Möglichkeit soll der Grundriss dabei in Echtzeit erstellt werden, um Änderungen am erzeugten Ergebnis zu erleichtern.

Der erzeugte Plan soll zunächst die Geometrie des als Grundlage genutzten Polygonnetzes genau widerspiegeln. Ist dies erreicht, soll versucht werden, den erzeugten Plan zu vereinfachen, indem die konkreten Positionen von Wänden und anderen geometrischen Merkmalen aus den Rohdaten ermittelt und in den erzeugten Plan übertragen werden.

4.5 Ergebnisexport

Der im vorhergehenden Schritt erstellte Grundriss soll zuletzt als weiter verarbeitbarer Plan exportiert werden. Dabei soll zunächst eine schlichte *Rastergrafik* erzeugt werden, welche den ermittelten Grundriss wiedergibt. Das konkrete Aussehen der Grafik sollte dabei vor dem Export, da sie auf sehr individuellen Eingabedaten basiert, entsprechend anpassbar sein, um eine optimale Darstellung zu erzielen. Die Grafik soll das .png-Format nutzen, welches als Standardformat weit verbreitet ist und zudem seinen Inhalt verlustfrei speichert, die eingetragenen Daten also unverändert wiedergibt.

Da eine solche Grafik recht ungenau und zudem schlecht anpassbar ist, ist sie für eine Nutzung in etablierten Arbeitsabläufen ungeeignet. Daher soll daraufhin versucht werden, aus den Grundrissdaten eine mit Prozessen des *Computer-aided Design* (CAD) kompatible Datei zu erstellen. Inhalte werden in einem solchen Format als mathematische Daten hinterlegt. So werden analog zu einer *Vektorgrafik* beispielsweise die Position und Ausrichtung einer Linie gespeichert, anstatt eine Abbildung der Linie als Pixelgrafik zu erstellen.

Die Daten einer solchen Datei sind mithilfe etablierter Anwendungen leicht anpassbar, wodurch beispielsweise bei der automatisierten Erstellung entstandene Fehler leicht berichtigt oder Vereinfachungen und/oder Erweiterungen des Plans effektiv umgesetzt werden können. Auch haben die Inhalte so erzeugter Pläne eine deutlich höhere Genauigkeit, da sie nicht durch die

Auflösung einer Grafik beschränkt sind und ihre konkrete Darstellung für einen besseren Überblick leicht angepasst werden kann.

Als Dateiformat soll dabei .dxf genutzt werden, da es sich dabei um ein Standardformat handelt, weches entsprechend gut von diverser Software unterstützt wird. Zudem ist es ein für Menschen leicht verständliches Format. Entsprechende Dateien könnten daher im Zweifel auch ohne Werkzeuge Dritter vergleichsweise leicht erstellt werden.

4.6 Bewertung der Resultate

Die letztlich durch das Projekt erzielten Ergebnisse sollen nach verschiedenen Gesichtspunkten beurteilt werden, um eine möglichst vollständige Bewertung des gesamten Prozesses zu erstellen. Über diese Aspekte wird im Folgenden ein kurzer Überblick gegeben.

Zunächst ist bezogen auf den Prozess der Planerstellung zu betrachten, welche Fehler oder Probleme abseits der Genauigkeit bei der Datenerfassung auftreten. Es sollen dabei regelmäßig fehlerhaft oder nicht erfasste Umgebungsmerkmale erläutert werden. Auch ist festzustellen, inwiefern diese Merkmale sich im letztlich erzeugten Grundrissplan widerspiegeln. So soll die Qualität der mit der HoloLens 2 erfassten Daten in Bezug auf den betrachteten Anwendungsfall dargelegt werden.

Weiterhin soll analysiert werden, wie vollständig die letztlich erzeugten Pläne sind. Dazu soll ein Vergleich mit üblichen Grundrissplänen anhand der jeweils dargestellten Inhalte erfolgen. So soll aufgezeigt werden, welche Merkmale tatsächlicher Grundrisspläne bereits in den automatisiert erstellten Plänen vorhanden und welche noch zu ergänzen sind.

Konkrete, mithilfe des dargelegten Prozesses erzeugte Grundrisspläne sollen daraufhin anhand von an ihnen ermittelten Maßen mit händisch erzeugten Plänen verglichen werden. So soll festgestellt werden, inwiefern die Maße des Plans sich mit den tatsächlichen Maßen decken beziehungsweise welche Abweichungen sich von der Realität ergeben. Explizit sollen dabei Wandstärken, Wandlängen und die Grundfläche verschiedener Räume abgeglichen werden, um eine Aussage zur Genauigkeit des Prozessresultats treffen zu können.

Zuletzt soll der gesamte, zur Planerstellung nötige Zeitaufwand sowie die Komplexität des Verfahrens beurteilt werden. Dafür ist das gesamte Vorgehen zu betrachten, welches für die Erstellung eines Grundrissplanes eines nicht erfassten Raumkomplexes erforderlich ist. So soll ein Überblick über den Arbeitsaufwand des Prozesses beziehungsweise seiner einzelnen Schritte gegeben werden.

5 Projektdurchführung

In den folgenden Abschnitten beschreibe ich den Ablauf der konkreten Ausführung des Projekts und erläutere die dabei letztlich in den Einzelschritten erreichten Zwischenergebnisse und Endresultate.

5.1 Erfassung und Export von Umgebungsdaten

Die Erstellung von Umgebungsdatensätzen mithilfe der HoloLens 2 stellte den grundlegenden Schritt bei der Projektausführung dar, da zur Umsetzung der weiteren Projektabschnitte konkrete Beispieldaten benötigt wurden. Zu Beginn konzentrierte ich mich dabei auf die Abwicklung des Datenexports mithilfe des Systems zur räumlichen Abbildung der HoloLens 2. Erst im späteren Projektverlauf probierte ich eine alternative Umsetzung mithilfe des Szenenverständnis-Systems, um für die Weiterverarbeitung besser geeignete Exportergebnisse erzielen zu können und zuvor aufgetretene Probleme nach Möglichkeit zu umgehen.

5.1.1 Nutzung des Systems zur räumlichen Abbildung

Zunächst testete ich die simpelste Exportmethode über das WDP, welches die Umgebungsdaten mit geringem Aufwand, verlässlich und ohne Fehler, beispielsweise fehlende Polygonnetzabschnitte, exportierte. Neben den bereits behandelten Nachteilen stellte ich dabei auch fest, dass diese Art des Exports keine Anpassungsmöglichkeiten in Bezug auf Detailgrad des Polygonnetzes sowie Größe des davon abgebildeten Bereichs bot und entsprechend unzureichende beziehungsweise ungenügende Daten bereitstellte.

Aus diesem Grund erstellte ich im Folgenden eine simple HoloLens-Anwendung mit Zugriff auf die vom System für räumliche Abbildung erzeugten Umgebungsdaten. Dies war durch die Konfiguration von im MRTK enthaltener, vorgefertigter Komponenten ohne großen Aufwand möglich, wobei unter anderem Einstellungen zu Detailgrad, Größe sowie Position des zu erfassenden Umgebungsbereichs und Aktualisierungsintervall getroffen werden konnten. Ein *Debugging* der Anwendungsausführung durch Ausgaben in der *Konsole* war dabei, statt direkt auf dem Gerät, über einen separaten Rechner, welcher sich im gleichen Netzwerk wie die HoloLens befand, möglich und wurde so im weiteren Projektverlauf für auf der HoloLens ausgeführte Programme genutzt.

Das so abgerufene Polygonnetz der Umgebung konnte zudem visualisiert werden, wobei die Kanten der Polygone über den zugehörigen Raumelementen dargestellt wurden und so ein

aktuelles Abbild der erfassten Daten lieferten. Ich setzte zudem eine Funktion zum Exportieren des Polygonnetzes in eine .obj-Datei um, wobei .obj ein Standardformat zum Speichern dreidimensionaler Objekte darstellt. Zum Starten dieses Prozesses fand dabei ein Sprachkommando Verwendung. Das erzeugte Polygonnetz war dabei grundsätzlich in würfelförmige Abschnitte mit einer Kantenlänge von 2,5 Metern unterteilt. Diese konnten wahlweise als einzelnes Netz zusammengefasst oder als separate, jedoch gruppierte Subnetze in der Exportdatei gespeichert werden.

Der Datenexport auf diese Art erwies sich ebenfalls als problematisch. So war die Bildfrequenz nach Starten des Exports sehr niedrig, bis die Bildausgabe der HoloLens 2 vollständig stoppte, was sich erst nach abgeschlossenem Export wieder änderte. Daher war es nicht möglich, Feedback zum aktuellen Stand des Vorgangs auszugeben.

Auch war die erzeugte Datei häufig unvollständig, wobei einige der Polygonnetzabschnitte fehlten, wie in Abbildung 5 zu sehen. Teilweise wurden zudem komplett leere Dateien erzeugt, ohne dass von der Anwendung oder der HoloLens 2 selbst Feedback bezüglich des fehlgeschlagenen Exports gegeben wurde.

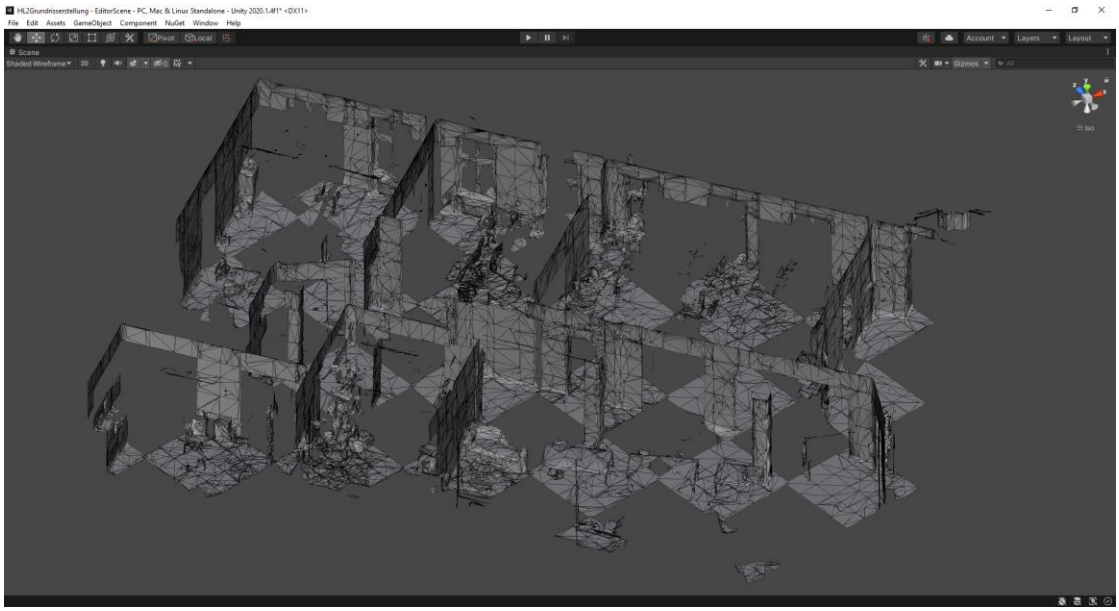

Abbildung 5: Von der HoloLens 2 fehlerhaft exportiertes Umgebungsnetz.

Zudem wurden, unabhängig von der gewählten Betrachtungsbereichsgröße, Netzabschnitte ab einer gewissen Distanz zum Zentrum des Betrachtungsbereiches nicht exportiert. Ich stellte fest, dass grundsätzlich ein Maximum von 128 Netzabschnitten zusammen exportiert werden konnte. Das Polygonnetz erschien dadurch abgeschnitten, wie in Abbildung 6 gezeigt. Durch mehrfaches Exportieren von verschiedenen Standpunkten aus konnten zwar alle Umgebungsdaten für größere Gebäudeabschnitte festgehalten werden, da bei den erzeugten

dreidimensionalen Modellen jedoch die Rotation und Positionierung relativ zu ihrem lokalen Koordinatenursprung nicht übereinstimmte, erwies sich ein Kombinieren dieser als nicht praktikabel. Das Größenlimit erwies sich also als starke Hürde bei der Arbeit mit größeren Räumlichkeiten.

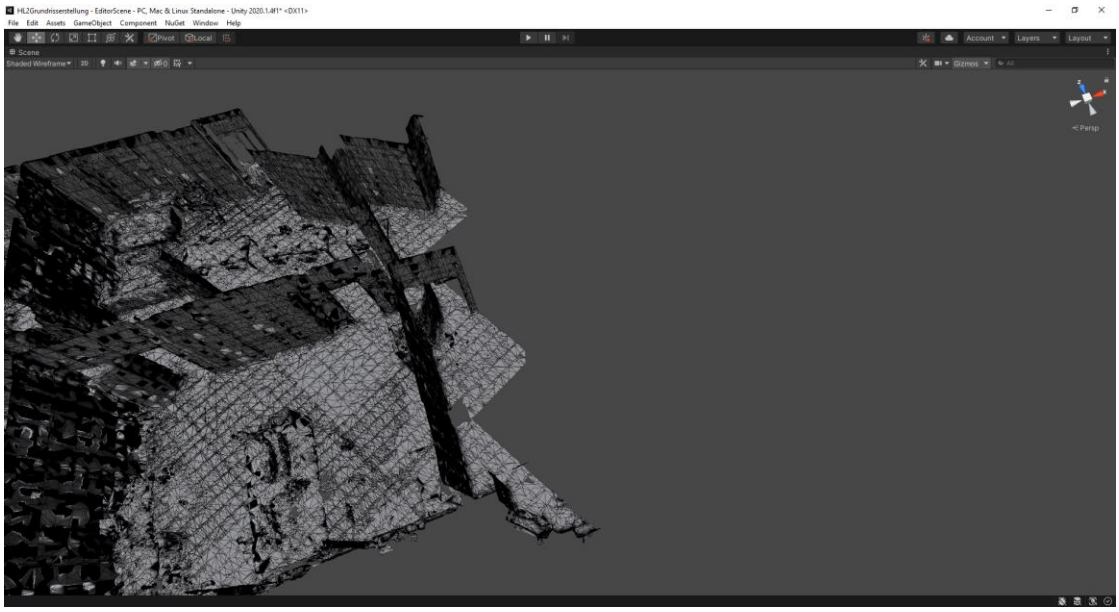

Abbildung 6: Exportiertes Umgebungsnetz mit abgeschnittenen Geometrien aufgrund der Größenbegrenzung des Systems zur räumlichen Abbildung.

Durch Anpassen verschiedener Einstellungen des MRTK sowie Änderungen am relevanten Programmcode konnte ich dabei keine Verbesserung der erwähnten Probleme herbeiführen. Der Export über das System zur räumlichen Abbildung erwies sich somit als zweckdienlich, jedoch nicht optimal für den projektrelevanten Anwendungsfall.

5.1.2 Nutzung des Szenenverständnis-Systems

Um die mit der ursprünglichen Exportmethode aufgetretenen Probleme zu umgehen, wandte ich mich im späteren Projektverlauf dem komplexeren Szenenverständnis-System als Exportmöglichkeit zu. Dabei konnte ich zur Einarbeitung in das System lediglich ein von Microsoft bereitgestelltes Beispielprojekt nutzen, da ansonsten, abseits einer Einleitung in die grundlegende Funktionsweise und Begrifflichkeiten, keine Dokumentation vorhanden war. Das Beispielprojekt ermöglichte dabei das Ausprobieren verschiedener Systemkomponenten sowie das Analysieren ihrer Implementierung innerhalb von Unity.

Erste Tests bestätigten mir, dass das bereitgestellte Polygonnetz im Szenenverständnis-System tatsächlich, abseits des Speichers der HoloLens 2, keine Größenbegrenzung hatte und damit

eines der bisherigen Probleme löste. Das Netz kann zudem mit einem deutlich höheren Detailgrad erstellt werden. Dabei besteht es grundsätzlich aus etwa gleichgroßen und gleichmäßig verteilten Polygonen. Anders als beim System zur räumlichen Abbildung werden bei weniger detaillierten Oberflächen, beispielsweise vollständig ebenen Böden, nicht weniger und/oder größere Polygone verwendet, wie in Abbildung 7 zu sehen. Entsprechend ist das erzeugte Polygonnetz zwar gleichmäßiger, jedoch weniger speichereffizient. Wie das vom System für räumliche Abbildung erzeugte Netz war es zudem ebenfalls in aneinander angrenzende Würfel mit einer Kantenlänge von 2,5 Metern unterteilt.

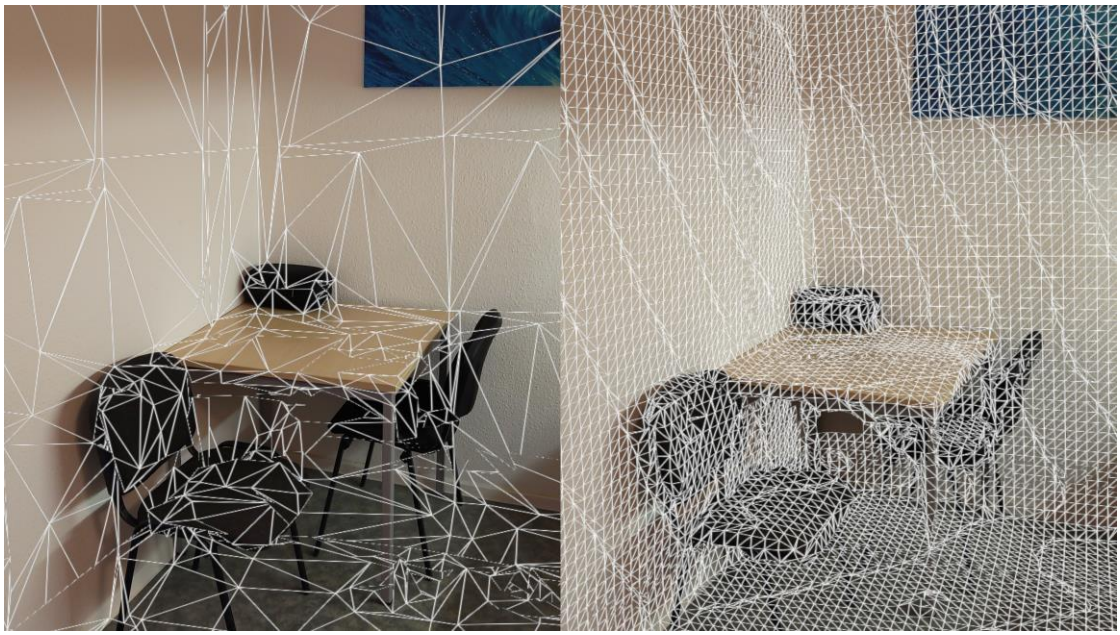

Abbildung 7: Vergleich der Polygonnetze des Systems zur räumlichen Abbildung (links) und des Szenenverständnis-Systems (rechts).

Analog zum vorherigen System setzte ich einen auf das Projekt zugeschnittenen Exportprozess um, welcher über einen Sprachbefehl ausgelöst werden konnte. Das Szenenverständnis-System stellte dabei die Umgebungsdaten in regelmäßigen Abständen zur Verfügung. Die zuletzt so erhaltenen Daten konnten dann in einer Anwendung abgerufen, visualisiert und exportiert werden, wobei beide Vorgänge die HoloLens 2 völlig auslasteten und kurzzeitig zu einem Ausbleiben der Bildausgabe führten und so das Geben von Feedback für anwendende Personen verhinderten. Beim Debugging lieferte die Anwendung dennoch weiterhin Ausgaben und konnte so zumindest ansatzweise überwacht werden.

Vor allem der Datenexport führte durch seine Dauer zu einem deutlich größeren Zeitaufwand als bei der ursprünglichen Methode. Diesen Zeitaufwand konnte ich durch Deaktivierung der Betrachtung der vom System interpretierten, vereinfachten Oberflächen verringern, wodurch nur noch das für das Projekt relevante Polygonnetz behandelt wurde. Das Problem, dass teilweise unvollständige Daten exportiert wurden, oder der Export fehlschlug und lediglich leere

Dateien erzeugte, setzten sich auch bei dieser Exportmethode fort, wobei keine entsprechenden Fehlermeldungen ausgegeben wurden.

Da zwischen den Aktualisierungen der Daten Zeitabstände von mehreren Sekunden lagen und auch die Visualisierung der aktuellen Daten nur nach einer kurzen Verzögerung geschah, entschied ich mich dafür, das aktuell erfasste Umgebungsnetz weiterhin über das System zur räumlichen Abbildung darstellen zu lassen. Das Szenenverständnis-System und sein besser geeignetes Polygonnetz fanden somit nur noch beim tatsächlichen Export Verwendung.

5.2 Extrahierung eines Grundrisses aus den erfassten Daten

Nach abgeschlossener Erfassung sowie Export von Beispieldaten, konkret eines möglichst kompletten, akkuraten Polygonnetzes eines Innenraumkomplexes, konnte ich mich nun dem nächsten Projektabschnitt zuwenden. Das Beispielnetz importierte ich dabei zunächst in Unity, wie in Abbildung 8 gezeigt. Dabei nutzte ich die Gelegenheit zur Überprüfung des Exportergebnisses. Nun sollte ein Prozess entwickelt werden, welcher einen Grundriss aus den so bereitgestellten Daten extrahiert und ihn nach Möglichkeit optimiert.

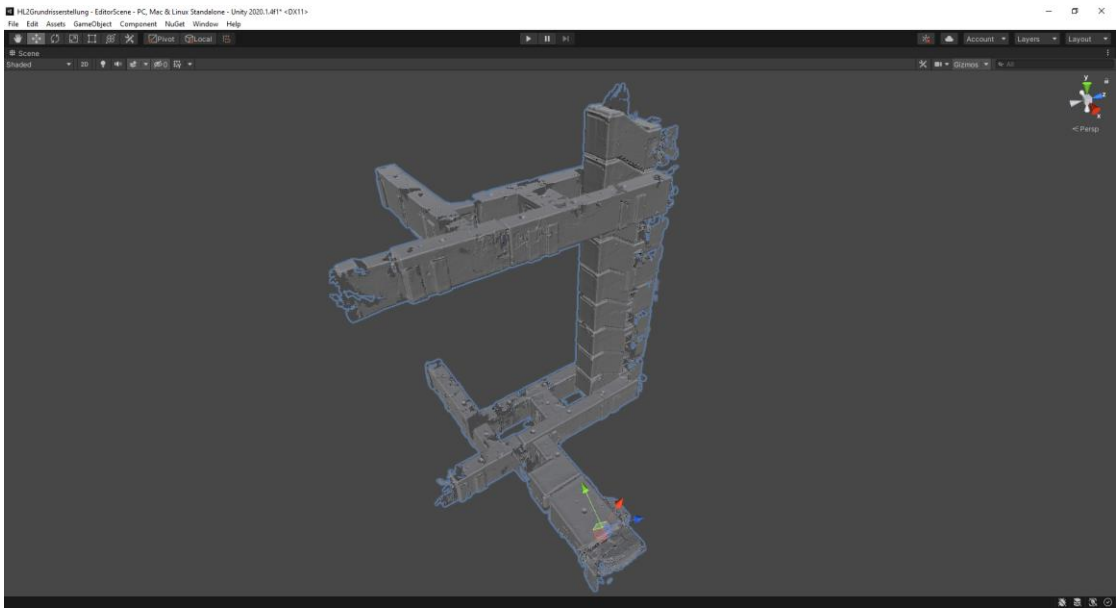

Abbildung 8: In Unity importiertes Umgebungsnetz.

5.2.1 Querschnitterstellung

Ich stellte zunächst fest, dass die in Unity bereitgestellten Polygonnetzdaten sich in einem für die Verarbeitung ungünstigen Format befanden. So stellt Unity eine Liste aller im Netz

befindlichen Eckpunkte beziehungsweise ihrer Positionen sowie eine Liste der Dreiecke des Netzes zur Verfügung. Die Dreiecksliste ist dabei konkret eine Auflistung der Indizes der die Dreiecke bildenden Eckpunkte, wobei jeweils drei aufeinanderfolgende Indizes auf die Eckpunkte eines Dreiecks verweisen.

Diese Datenstrukturen entsprechen dabei nicht den von Unity intern Verwendeten, sondern dienen rein dem Zugriff und der Bearbeitung in erstellten Anwendungen. Daher werden die beiden Listen auch bei jedem Abruf rechenaufwändig neu erstellt. Da aufgrund dieser Tatsache ein Festhalten der Daten in eigenen Listen notwendig war, entschied ich mich, die Datenstruktur bei dieser Übertragung anzupassen, um ein angenehmeres Arbeiten mit ihr zu ermöglichen. Die Eckpunktliste ließ ich dabei unberührt und änderte lediglich die Dreiecksauflistung zu einer Liste an Dreiecksobjekten, welche die Eckpunktindizes des jeweiligen Dreiecks beinhalteten. So konnten später zusätzlich weitere für die Verarbeitung relevante Informationen zu den Dreiecken in den jeweiligen Objekten gespeichert werden. Diesen Prozess der Datenabfrage und -aufbereitung gestaltete ich aufgrund des mit ihm verbundenen Rechenaufwands dabei so, dass er einmalig beim Programmstart und danach nur auf Wunsch der die Anwendung nutzenden Person ablief.

Nach dieser Vorbereitung sollten in der Folgenden ersten Phase der tatsächlichen Querschnittserstellung nun die Schnittkanten der im Polygonnetz befindlichen Dreiecke mit einer Fläche ermittelt werden. Die Anwendung überprüfte dafür bei jedem Dreieck, ob einer der Eckpunkte auf einer anderen Seite der Schnittfläche liegt als die Anderen. War dies der Fall, wurden die Schnittpunkte der Dreieckskanten mit der Fläche berechnet, welche Start- beziehungsweise Endpunkt der Schnittkante des Dreiecks darstellten.

Die Fläche ließ ich dabei zur besseren Veranschaulichung durch ein tatsächliches Flächenobjekt in der Szenenansicht von Unity repräsentieren. Dadurch konnte die Schnittfläche von der Person, welche die Anwendung verwendet, frei bewegt und der erzeugte Querschnitt so angepasst werden. Diese Verwendung der Szenenansicht bot sich anstelle der aufwändigeren Entwicklung einer eigenen Interaktionsschnittstelle in der Spielansicht an, da die Lauffähigkeit der Anwendung außerhalb des Editors für das Projekt nicht von Priorität war. Somit wurde die gesamte, in Abbildung 9 gezeigte Anwendung für die Nutzung innerhalb des Unity-Editors erstellt.

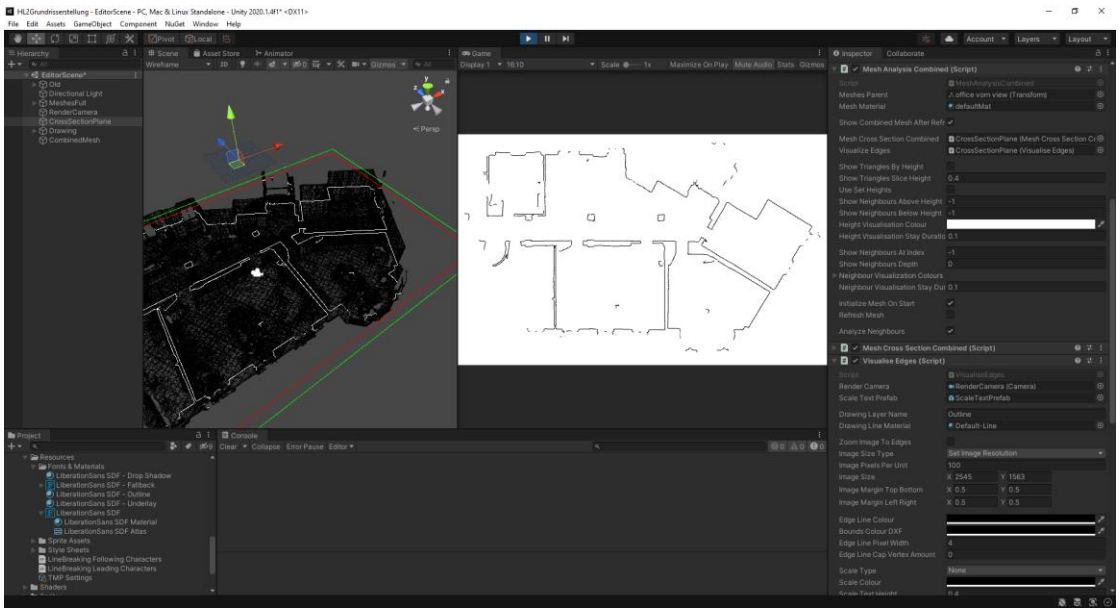

Abbildung 9: Anwendung zur Querschnittsextrahierung im Unity-Editor.

Die erzeugten Schnittkanten stellte ich zunächst in der Szenenansicht als, üblicherweise für Debugging-Visualisierungen verwendete, nur in dieser Ansicht wiedergegebene Debugging-Linien dar, welche so direkt am Polygonnetz abgebildet wurden. Abbildung 10 zeigt die daraus resultierende Darstellung. In der Hauptansicht erfolgte die Visualisierung durch sogenannte Linienrenderer, Objekte, welche dreidimensionale Linien mit anpassbarem Aussehen darstellen. Dabei wurden alle anderen Objekte, also das Polygonnetz sowie das die Schnittfläche repräsentierende Objekt, ausgeblendet. Die verwendete Kamera passte ich so an, dass sie eine parallelprojizierte Draufsicht auf die erzeugten Linien darstellte und ließ ihr Sichtfeld automatisch so anpassen, dass alle erzeugten Linien in diesem lagen.

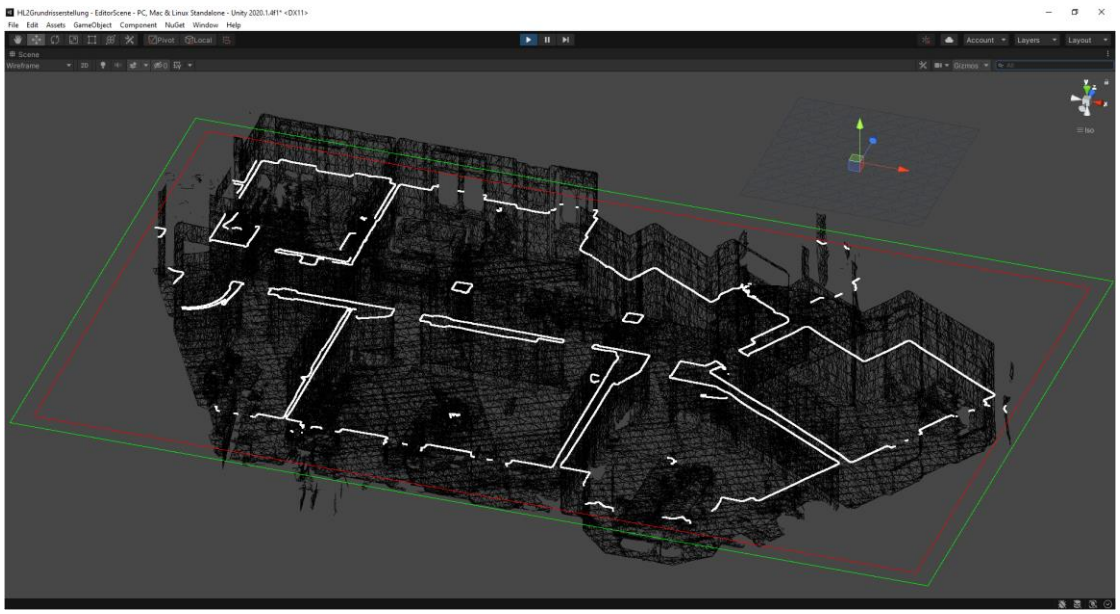

Abbildung 10: Visualisierung des Querschnitts am Polygonnetz.

Da die Dreiecke des Polygonnetzes nicht nach Position oder ähnlichen Eigenschaften geordnet vorlagen, mussten für diesen Prozess der Schnittkantenfeststellung zunächst alle Dreiecke auf einen Schnitt mit der Fläche untersucht werden, was einen hohen Rechenaufwand mit sich brachte. Ließ man den Prozess in Echtzeit ablaufen, waren Interaktionen mit dem Programm kaum mehr möglich, da die Anwendung beim Warten auf die Fertigstellung des Berechnungsprozesses keine Eingaben entgegennahm.

Die erzeugten Schnittkanten waren zusammen als Grundriss erkennbar, wie in Abbildung 11 beispielhaft dargestellt. Die gewählte Methode bot sich also als Grundlage für die Weiterentwicklung an. Der aktuelle Zustand der Anwendung bedurfte allerdings Optimierungen, um eine tatsächliche Verwendung zu ermöglichen.

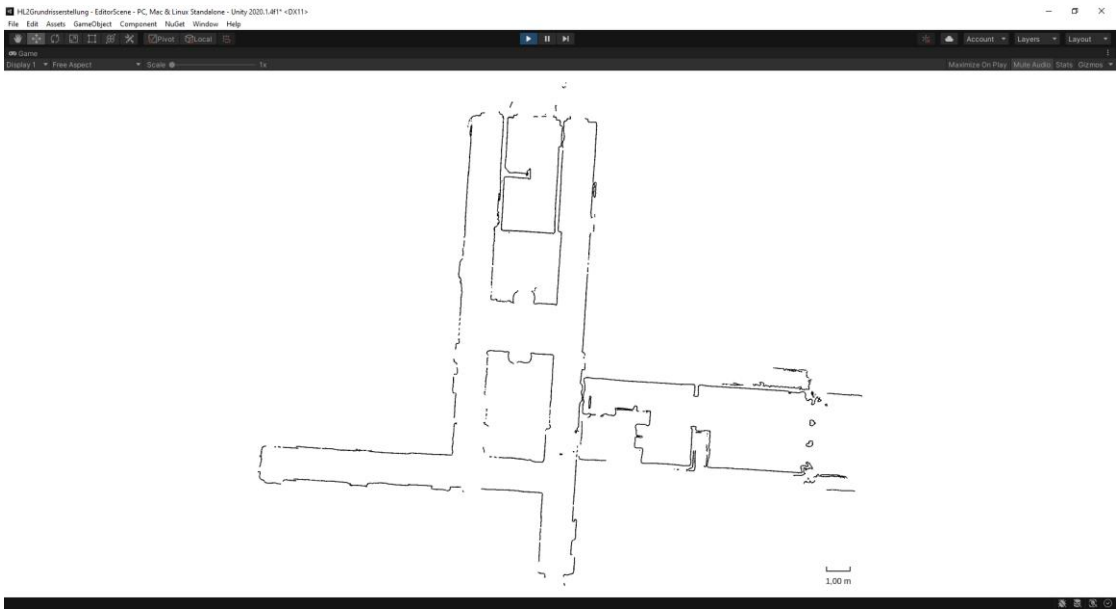

Abbildung 11: Vorschau des Querschnittsplans, welcher mit den aktuellen Einstellungen erstellt werden würde.

5.2.2 Optimierung des Prozesses

Im Folgenden Projektverlauf beschäftigte ich mich zunächst damit, eine in Echtzeit ablaufende Interaktion mit der Anwendung zu ermöglichen. Aktuell wurde für jedes ausgegebene Bild zunächst der Querschnitt des gesamten Polygonnetzes festgestellt. Dieser langwierige Prozess würde dabei mit zunehmender Größe des Polygonnetzes immer länger dauern und konnte somit, in Bezug auf die benötigte Rechenzeit, trotz Optimierungen im Prozess ab einer gewissen, erfassten Datenmenge wieder problematisch werden. Da die Anwendung komplett im Unity-Editor ausgeführt wurde, konnten zudem die für seine Ausführung nötigen Systemressourcen nicht zur Beschleunigung des Programmablaufes genutzt werden, was abseits der Entwicklung einer alleinstehenden Anwendung jedoch nicht verhindert werden konnte.

Aus diesem Grund passte ich den Ablauf so an, dass die Berechnung der Schnittkanten über eine beliebig einstellbare Anzahl an ausgegebenen Bildern hinweg ausgeführt wurde. Durch Erhöhung dieser Anzahl konnte so eine angenehme Interaktion ermöglicht werden, Änderungen an der Ausrichtung der Schnittfläche wurden aber nur noch mit einer entsprechenden Verzögerung der gewählten Bildanzahl umgesetzt. Durch Senken der Aktualisierungsgeschwindigkeit konnte nun also unabhängig von der betrachteten Datenmenge ein angenehmes Nutzungserlebnis ermöglicht werden.

Nach dieser Änderung begann ich zudem den für die Schnittkantenberechnung zuständigen Algorithmus anzupassen. Im Moment waren dabei viele der betrachteten Dreiecke nicht in der Nähe der Schnittfläche und konnten für die Berechnung entsprechend ignoriert werden. Da für den zu erstellenden Grundrissplan lediglich die Draufsicht auf das Polygonnetz relevant war, musste die Schnittfläche zudem immer nur horizontal im Raum liegen. Daher war für die Schnittberechnung lediglich die Höhe, in welcher die Dreiecke lagen, relevant.

Um entsprechende Optimierungen durchzuführen, war es zunächst erforderlich, eine Aussage zur vertikalen Position jedes Dreiecks treffen zu können. Ich passte dazu die Erstellung der Liste an Dreiecksobjekten an, sodass in den Objekten zusätzlich die Höhe des niedrigsten und höchsten Eckpunkts des Dreiecks, sowie die mittlere Höhe, also der Durchschnitt dieser beiden Werte, gespeichert wurde. Die Liste ließ ich dabei nach der mittleren Höhe sortieren.

Daraufhin ermöglichte ich es, bei der Anwendung einen Betrachtungsbereich unter und über der Schnittfläche festzulegen. Bei der Schnittkantenberechnung wurden nun nur noch Dreiecke, deren mittlere Höhe innerhalb dieses Bereichs lag, betrachtet. Je kleiner dieser Bereich gewählt wird, desto weniger Dreiecke muss der Algorithmus bearbeiten, wodurch er schneller abläuft. Ist der Bereich zu klein, kann es jedoch dazu kommen, dass Dreiecke, welche eigentlich die Fläche schneiden, aufgrund ihrer konkreten Lage nicht betrachtet werden. Optimalerweise sollte die Größe des Bereichs so gewählt werden, dass sie etwas größer als die Größe der Dreiecke im Netz ist, um keine Schnittkanten auszulassen und dennoch möglichst wenige Dreiecke in die Betrachtung einzuschließen.

Da nun nur noch die vertikale Position der Schnittfläche und damit die Höhe der Dreiecke beziehungsweise ihrer Eckpunkte für die Schnittberechnung betrachtet wurde, konnte diese zudem vereinfacht werden. So konnte nun, um festzustellen, ob ein Eckpunkt über oder unter der Fläche lag, einfach die Höhe der Fläche mit der des Eckpunktes verglichen werden, was den Rechenaufwand deutlich verringerte.

5.2.3 Ansatz zur Vereinfachung des erzeugten Querschnitts

Nachdem nun ein schlichter Querschnitt aus dem Polygonnetz erstellt werden konnte, sollten im Folgenden Möglichkeiten betrachtet werden, den erzeugten Plan zu vereinfachen, indem zu einem geraden Wandabschnitt gehörende Linien als eine Gesamtlinie dargestellt werden würden. So würde der Plan weniger unruhig erscheinen und es böte sich die Möglichkeit an, die Längen einzelner Wandabschnitte im Plan zu vermerken.

Die so erreichte Verringerung der darzustellenden Linien hätte zusätzlich eine Leistungsverbesserung bei der Anwendung bewirkt. Dazu sollten konkret Schnittkanten, deren zugehörige Dreiecke zu einem Wandabschnitt gehören, zusammengefasst werden, um die konkret abgebildeten Gebäudemerkmale zu extrahieren. Um dies zu erreichen, sollten zuerst im Plan Ecken festgestellt werden, welche daraufhin zu Wänden verbunden werden würden.

Um auf diese Weise zusammenhängende Wandabschnitte ermitteln zu können, war es zunächst notwendig, Kenntnis über die Aneinanderreihung der berechneten Schnittkanten zu haben. Da die zu aneinandergrenzenden Schnittkanten gehörigen Dreiecke ebenfalls aneinandergrenzen mussten, war also grundsätzlich für jedes Dreieck festzustellen, welche anderen Dreiecke an seinen Kanten anliegen. Da die Dreiecke im zur Speicherung des Polygonnetzes genutzten Datenformat nicht entsprechend geordnet sind und Unity kein System zur Ermittlung solcher Nachbarbeziehungen anbot, mussten diese eigenständig festgestellt werden.

Ich passte also den Datenaufbereitungsprozess des Polygonnetzes entsprechend an, sodass er in jedem Dreiecksobjekt dessen Nachbardreiecke vermerkte. Dabei ließ ich die Eckpunkte jedes bearbeiteten Dreiecks mit denen aller bereits Abgearbeiteten vergleichen. Stimmten zwei der Eckpunkte überein, handelte es sich um benachbarte Dreiecke und beide Dreiecksobjekte erhielten einen Verweis auf das jeweils Andere. Die so ermittelten Nachbarbeziehungen konnten später potentiell zusätzlich für weitere Optimierungen beim Extrahieren des Grundrisses genutzt werden. Abbildung 12 zeigt die so von einem einzelnen Dreieck ausgehenden Nachbarbeziehungen.

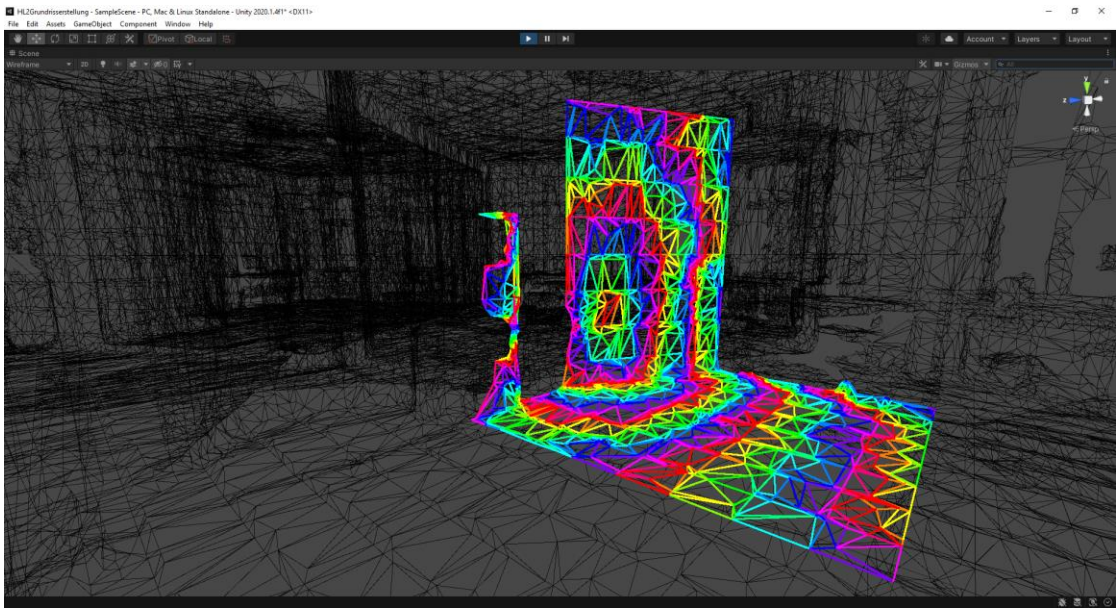

Abbildung 12: Visualisierung der ermittelten Nachbarschaftsbeziehungen zwischen den Dreiecken des Umgebungsnetzes ausgehend von einem zentralen Dreieck.

Während der Umsetzung beziehungsweise beim Testen dieser Anpassung stellte sich dabei ein Problem heraus. Die würfelförmigen Abschnitte, in welche die exportierten Polygonnetze unterteilt waren, schlossen nicht, wie angenommen, sauber miteinander ab, sondern überlappten an ihren Grenzen um mehrere Zentimeter, wie in Abbildung 13 dargestellt. Über die Abschnittsgrenzen hinweg konnte somit, selbst wenn alle Netzabschnitte kombiniert wurden, keine direkten Nachbarbeziehungen festgestellt werden.

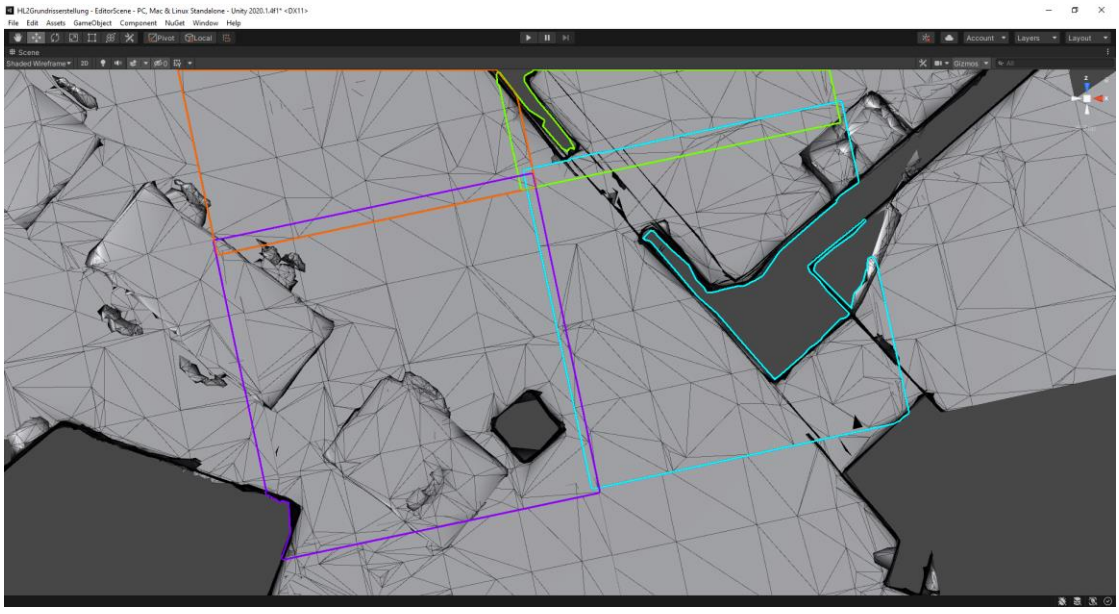

Abbildung 13: Überlagerung der würfelförmigen Abschnitte des Polygonnetzes.

Entsprechend konnten zunächst auch nur in einem Abschnitt aneinandergrenzende Querschnittslinien miteinander verbunden werden. Dies funktionierte dabei ohne Probleme, wobei pro Netzabschnitt ein oder, bei Lücken im erfassten Netz oder mehreren nicht miteinander verbundenen Wänden, welche den Abschnitt durchlaufen, mehrere geordnete Listen der jeweils zusammenhängenden Schnittkanten ermittelt wurden.

Die so erzeugten Abschnitte ließ ich daraufhin bestmöglich automatisiert miteinander verbinden. Dabei wurde die Distanz aller Start- und Endpunkte der ermittelten Kantenabschnitte ermittelt und jeweils die zwei Punkte mit der niedrigsten Distanz, sofern diese unter einem festgelegten Schwellwert lag, als miteinander verbunden angesehen. So wurden erneut geordnete Listen der jeweils zusammenhängenden Kantenabschnitte, wie beispielhaft in Abbildung 14 gezeigt, erstellt und eine insgesamt möglichst akkurate Reihenfolge aller zusammenhängenden Schnittkanten ermittelt.

Abbildung 14: Unterteilung der ermittelten Querschnittkanten in zusammenhängende Kantenabschnitte.

Ich ließ zudem Kantenabschnitte innerhalb einzelner Netzabschnitte miteinander verbinden, um kleinere, nicht erfasste Bereiche zu schließen. Auf die gleiche Art ließ ich die Kantenabschnitte auch über die Grenzen der Netzabschnitte hinweg zusammenfügen. Beispielhaft sind so erzeugte Kantenabschnitte in Abbildung 15 dargestellt. Das Distanzlimit für die Verbindung verschiedener Kantenabschnitte sollte für den Prozess dabei möglichst klein gewählt werden. War es beispielsweise größer, als die Dicke der erfassten Wände, wurden Kantenabschnitte häufig inkorrekt durch die Wände hindurch verbunden.

Abbildung 15: Zusammenfassung ermittelter Kantenabschnitte deren Enden nah beieinander liegen. Erzielte in diesem Fall zur Verringerung der Abschnittsanzahl von 189 auf 66.

Aus Zeitgründen war es mir nicht möglich, diesen Vereinfachungsansatz weiter zu verfolgen. Im weiteren Verlauf hätten aber die Schnittkanten innerhalb der ermittelten Kantenabschnitte nacheinander abgearbeitet werden sollen. Zwischen den jeweils angrenzenden Kanten hätte daraufhin der kleinste Winkel festgestellt werden müssen, wobei Winkel unter einer festgelegten Größe als Eckpunkte markiert worden wären. Die zu einem Abschnitt gehörenden Eckpunkte wären daraufhin miteinander verbunden worden.

Auch boten sich einige Optimierungsmöglichkeiten, mit denen ich mich nicht mehr beschäftigen konnte. So hätte die für den Prozess ungünstige Unterteilung des Polygonnetzes potentiell genutzt werden können, um die Berechnungsgeschwindigkeit einzelner Abläufe zu beschleunigen. Beispielsweise hätte durch Zuordnung aller Dreiecke zu ihrem jeweiligen Netzabschnitt die Ermittlung der Nachbarschaftsbeziehungen vereinfacht werden können. Statt alle Dreiecke des Polygonnetzes als potentielle Nachbarn zu betrachten, hätten dabei solche aus anderen Abschnitten direkt ausgeschlossen und die Berechnungszeit signifikant verringert werden können.

In der umgesetzten Version werden zudem bei der Verbindung der Kantenabschnitte die Start- und Endpunkte aller Abschnitte mit allen Übrigen, noch nicht an beiden Seiten in einem Verbund Befindlichen, verglichen, was gerade bei großen Mengen an Kantenabschnitten eine sehr hohe Laufzeit mit sich bringt. Eine Beschränkung dieses Vergleichs auf Kantenabschnitte angrenzender Netzabschnitte wäre jedoch deutlich effektiver. Dazu müsste die Lage jedes Abschnitts des Polygonnetzes beziehungsweise der darin befindlichen Dreiecke relativ zu den

anderen Abschnitten festgestellt und damit effektiv in ein dreidimensionales Raster der Abschnitte eingeordnet werden.

5.2.4 Alternative Ansätze für die Grundrisserstellung

Einige weitere Ansätze, welche für die Erstellung des Grundrisses aus dem Polygonnetz in Betracht gezogen worden waren, wurden aus Zeit- oder Praktikabilitätsgründen nicht genutzt oder nur testweise in grundlegender Form umgesetzt. Diese Alternativen und die Gründe, wegen derer sie nicht weiterverfolgt wurden, erläutere ich im Folgenden kurz.

Zwischenzeitlich stand die Möglichkeit im Raum, statt der für die Querschnittserstellung genutzten Schnittfläche einen Schnittkörper zu verwenden. Dabei sollten alle im Bereich des Schnittkörpers liegenden Dreieckskanten festgestellt und in den Plan übertragen werden, wodurch vom Polygonnetz nicht erfasste Bereiche auf einer Höhe durch auf einer anderen Höhe erfasste Daten ausgeglichen werden. Auch sollten Strukturen, welche nur in bestimmten Höhen auftraten, besser im erstellten Plan widergespiegelt werden.

Nachdem die Höhe der Dreiecke des Polygonnetzes für die Schnittanalyse mitbetrachtet wurde, konnte anhand der so verfügbaren mittleren Höhe leicht, wenngleich nicht ganz genau, festgestellt werden, ob ein Dreieck sich innerhalb eines solchen Schnittkörpers befand. Ich setzte den Ansatz daher testweise um, um mir ein Bild des erzielten Ergebnisses, in Abbildung 16 dargestellt, zu verschaffen. Letztlich entschied ich mich dafür, die Schnittkörpermethode im Rahmen dieses Projekts nicht weiter zu betrachten, da sie ein weniger sauberes, unruhiges Ergebnis erzeugte, wie in Abbildung 17 vergleichsweise gezeigt, und die erhaltenen Daten sich als ungeeignet für eine schlichte Auswertung zur Vereinfachung des Querschnitts herausstellten.

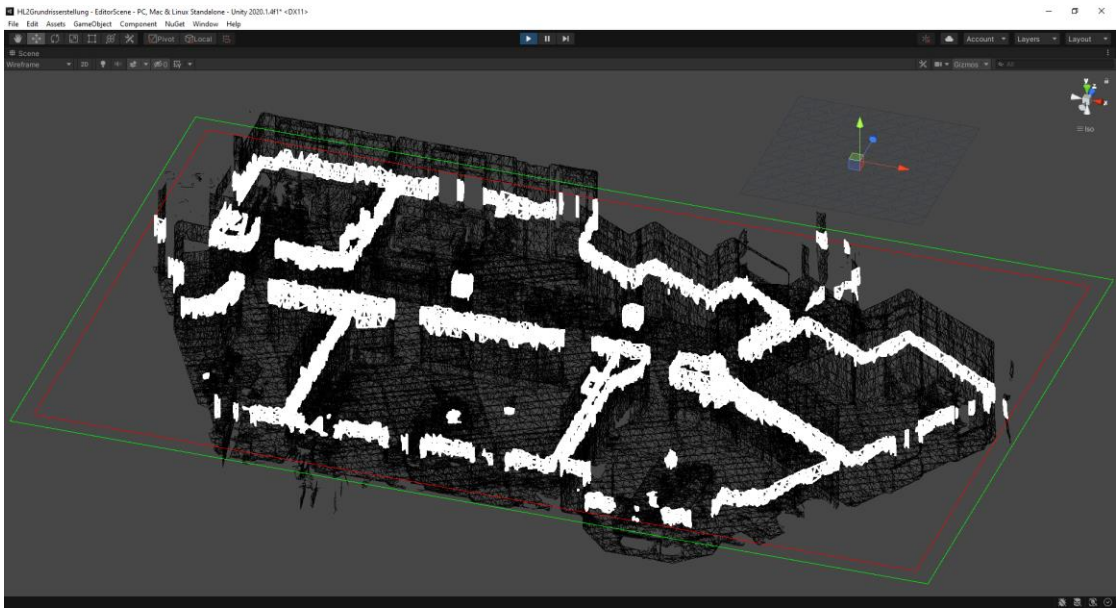

Abbildung 16: Visualisierung des Querschnitts bei Verwendung eines Schnittkörpers.

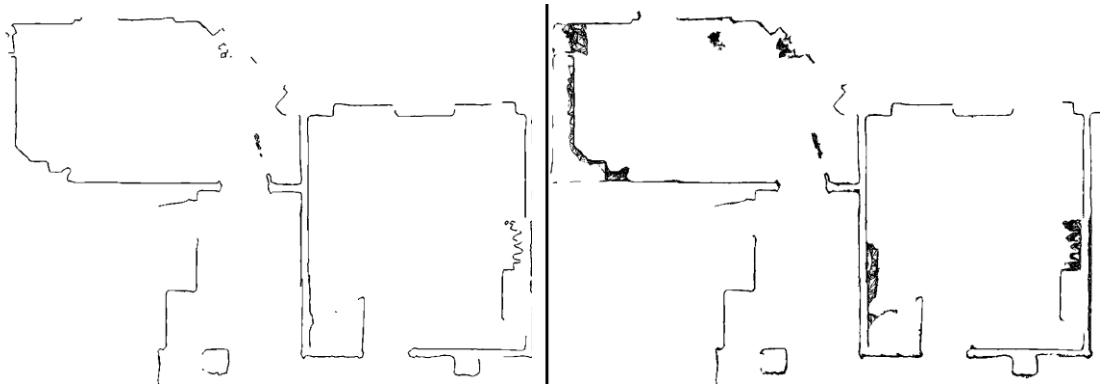

Abbildung 17: Vergleich eines mithilfe einer Schnittfläche erzeugten Querschnitts links mit einem unter Verwendung eines Schnittkörpers Erzeugten rechts.

Ein anderer Ansatz, um Grundrissdaten zu erhalten, sollte eine Parallelprojektion des Polygonnetzes von oben oder unten her darstellen. Da die Wände der abgebildeten Gebäudestruktur frei von jeglichen Polygonen sein sollten, würden diese nicht in der Projektion widergespiegelt. Die Böden, Decken und Wandaußenseiten der Räume sollten hingegen komplett von Polygonen bedeckt und somit in der Projektion erfasst sein.

Problematisch wäre es bei Verwendung dieser Methode gewesen, dass höhenspezifische Strukturen, beispielsweise Fenster, sich nicht im Ergebnis der Projektion wiedergefunden hätten. Auch wäre es nötig gewesen, die Projektion auf einen festgelegten vertikalen Bereich zu beschränken, sofern die Umgebungsdaten mehrere Etagen mit unterschiedlichen Raumanordnungen abbildeten. Bei Einbeziehung mehrerer Etagen in die Projektion wäre es so zu falschen Ergebnissen gekommen, da polygongefüllte Bereiche aus einer Etage polygonleere Abschnitte einer anderen überdeckt hätten. Auch wäre die Ableitung von Wandabschnitten oder

bestimmten Strukturen aus dem in diesem Verfahren erzeugten Ergebnis nicht trivial. Die erhaltenen Daten wären für den gewünschten Verwendungszweck also deutlich ungünstiger nutzbar.

Aus diesen Gründen, im Zusammenspiel mit der guten Funktionsweise der auf einem Querschnitt basierenden Methode, wurde der Ansatz frühzeitig verworfen und nicht ausprobiert.

5.3 Ergebnisexport

Die im vorherigen Schritt extrahierten Schnittkanten beziehungsweise der von ihnen gebildete Querschnitt sollten im Folgenden aus der genutzten Unity-Anwendung exportiert werden. Dabei sollte durch Verwendung eines geeigneten Exportformates letztlich eine Nutzung und/oder Weiterverarbeitung des Planes außerhalb des zur Erstellung verwendeten Programms ermöglicht werden.

5.3.1 Export als Rastergrafik

Zunächst sollte aus den Kanten des zuvor generierten Querschnitts eine Rastergrafik erzeugt werden. Unity erlaubt es, zur Programmlaufzeit Texturen zu erstellen, die Farbe ihrer einzelnen Pixel festzulegen und sie dann als Grafiken zu exportieren. Auf dieser Grundlage könnte man Systeme zum Einzeichnen von Linien oder ähnlichen grafischen Elementen umsetzen. Da die Rastergrafik jedoch nur als temporäre Exportvariante umgesetzt werden sollte, war dieses Vorgehen aufgrund des damit verbundenen Zeitaufwandes, gerade für die Implementierung komplexerer Elemente, beispielsweise Text, für das Projekt ungeeignet.

Ich entschied mich daher dafür, diese Methode zunächst zurückzustellen und stattdessen eine sogenannte Rendertextur als schneller umsetzbare, wenngleich weniger präzise auf den vorliegenden Anwendungsfall abgestimmte Möglichkeit für den Export zu nutzen. Rendertexturen in Unity sind Texturen, welche die aktuelle Ansicht einer in der Szene befindlichen Kamera als Grafik speichern können. Da alle Schnittkanten bereits als Linienrenderer in der Szene dargestellt und von einer Kamera komplett eingefangen wurden, konnte ich also schlicht das Bild dieser Kamera in einer Rendertextur speichern und dann als Rastergrafik exportieren. Ein großer Vorteil dieser Methode ist es, neben dem geringen Umsetzungsaufwand, dass das so exportierte Bild, eventuell abgesehen von seiner Auflösung, genau dem in der Anwendung dargestellten Plan entsprechen würde.

Ich setzte diesen Ansatz basierend auf der bereits bestehenden Szenenstruktur um, musste also lediglich die Abläufe zum Speichern und Exportieren des Bildes ergänzen. Zudem fügte ich Einstellungsmöglichkeiten für Bildgröße oder -auflösung hinzu und implementierte Optionen zur Änderung von Linienstärke und -farbe sowie des Zuschnitts des erzeugten Plans, um sein konkretes Aussehen anpassen zu können.

Auch wurde die wahlweise Platzierung der Kamera anhand der Gesamtgröße des Polygonnetzes oder des ermittelten Querschnitts realisiert. Dabei ermöglichte die erstere Variante das Verschieben der Schnittfläche ohne ein, durch die Größenänderung des Querschnitts bedingtes, starkes Schwanken des gezeigten Bildausschnitts und bot sich daher besser zur Anpassung der Querschnittshöhe an. Die zweite Möglichkeit minimierte hingegen den im Bild befindlichen Weißraum, war also für die Anpassung des Bildausschnitts für den Export besser geeignet.

Zuletzt fügte ich eine Maßstabsangabe ein, welche die Größe einer festgelegten Distanz im Plan durch eine beschriftete Linie darstellte. Dies war leicht möglich, da im MRTK ein Meter der realen Welt einer Längeneinheit von Unity entspricht [18] und dieser Umrechnungsfaktor sich entsprechend auch beim von der HoloLens 2 exportierten Polygonnetz wiederfand. Die Position der Maßstabsangabe sowie die Formatierung von Text und Linien machte ich dabei analog zum Querschnitt anpassbar. An sich war die so umgesetzte Maßangabe dabei wenig aussagekräftig und nicht zwingend nötig, diente aber zum Test für die eventuell zu realisierende Bemaßung des Plans benötigter Konzepte, konkret der Arbeit mit Text und der maßstabsgerechten Skalierung von Planelementen.

Die so erstellten .png-Grafiken stellten sich als insgesamt sehr gute Möglichkeit zur Überprüfung der aktuellen Funktionsweise der Anwendung und zum Vergleich von Ergebnissen verschiedener Prozessiterationen heraus. Abbildung 18 zeigt ein so exportiertes Bild.

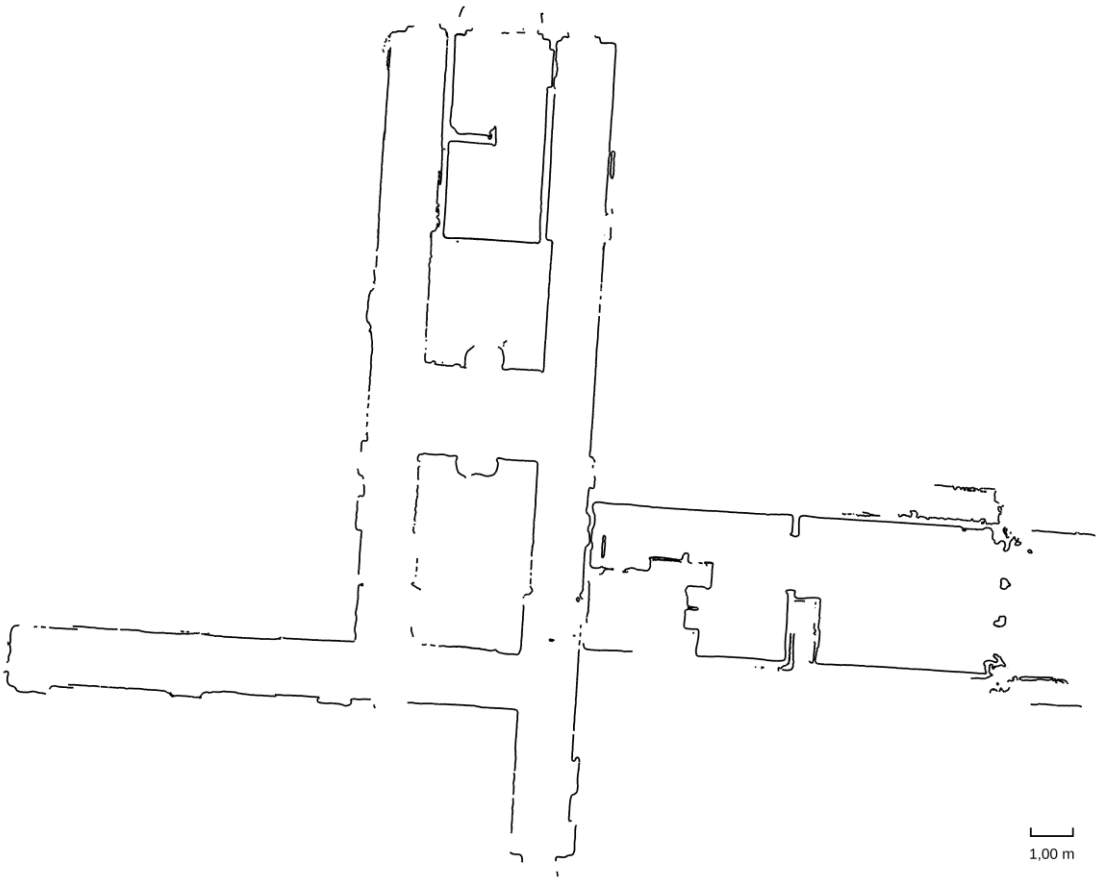

Abbildung 18: Als .png-Datei exportierter Querschnittsplan.

5.3.2 Export als CAD-kompatible Datei

Nachdem das Exportieren von Rastergrafiken nun möglich war, wendete ich mich der Erstellung von den Plänen entsprechenden .dxf-Dateien zu. Dabei suchte ich zunächst nach einer Programmbibliothek, welche diesen Prozess erleichtern sollte, da die Umsetzung eines eigenen Systems zwar aufgrund des simplen Aufbaus der Dateien möglich, jedoch sehr zeitaufwändig gewesen wäre.

Ich stieß dabei auf verschiedene für das Projekt geeignete Bibliotheken, legte mich letztlich aber auf „netDxf“ fest, da diese kostenlos zur Verfügung gestellt und unter einer LGPL-Lizenz angeboten wurde. Durch dieses Lizenzmodell kann eine Programmbibliothek frei in Anwendungen genutzt werden, solange für das Programm nutzende Personen die Möglichkeit besteht, den in der Bibliothek bereitgestellten Programmcode anzupassen. [19] Alternative Programmbibliotheken waren hingegen entweder direkt in Bezahlmodellen verfügbar oder ihre Verwendung war an Lizenzen für CAD-Anwendungen geknüpft und durch den so entstehenden Kostenaufwand nicht für das Projekt geeignet.

Ich importierte netDxf in das Projekt und führte zunächst einige Tests der enthaltenen Funktionen durch. Die Umsetzung dieser Tests gelang dabei schnell und die erzeugten Dateien waren tatsächlich von gängigen CAD-Anwendungen verwertbar. Entsprechend übertrug ich im Folgenden den aktuellen Stand des Rastergrafik-Exports, also die Darstellung von Querschnitt und Maßangabe, in einen Prozess zum Exportieren einer .dxf-Datei. Dabei stellte sich gerade die Übertragung des an der Maßangabe befindlichen Textes als nicht trivial, jedoch gut umsetzbar heraus. Das Aussehen des Textes wich dabei, anders als bei den Rastergrafiken, von dem in der Vorschau gezeigten Stil ab, was für die Planfunktion jedoch unerheblich war.

Somit konnten den Rasterbildern äquivalente Dateien erstellt werden, welche in gängigen CAD-Anwendungen nutzbar und damit leicht anpassbar waren. Abbildung 19 zeigt ein Beispiel für eine so erstellte Datei. Den Export als .png-Datei behielt ich in der Anwendung als Option bei, da mithilfe der von ihm erzeugten Grafiken ein einfacheres Überprüfen des aktuellen Planaussehens möglich und eine höhere Kompatibilität mit anderen Programmen gegeben war.

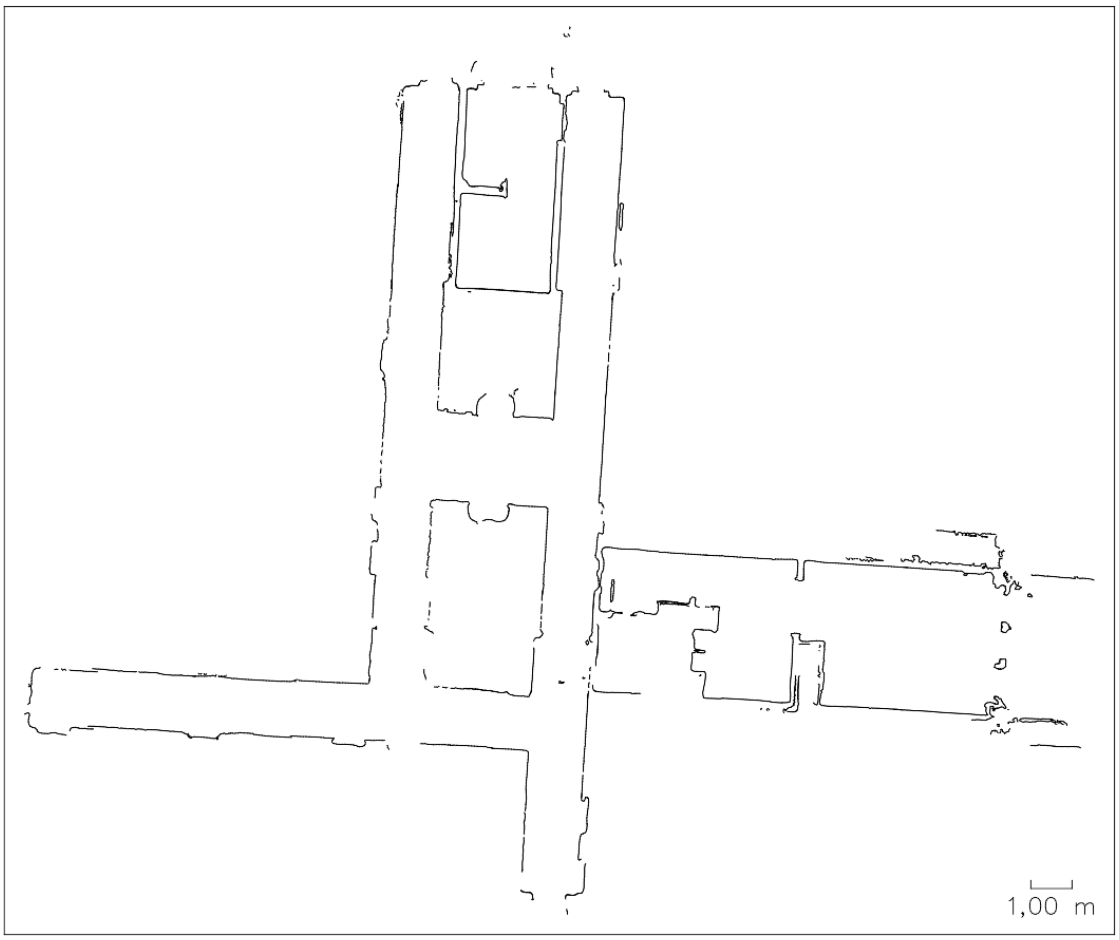

Abbildung 19: Als .dxf-Datei exportierter Querschnittsplan mit den Inhalt einschließendem Rahmen und anderer Schriftart.

6 Ergebnisbewertung

Nachdem das Projekt umfassend bearbeitet und die konkrete Ausführung aller Einzelschritte ermittelt wurde, wandte ich mich der Beurteilung des Prozessablaufs sowie der erzielten Ergebnisse zu. Über diese Auswertung entsprechend der zuvor festgelegten Kriterien gebe ich im Folgenden einen Überblick.

6.1 Qualität der erfassten Daten

In den von der HoloLens 2 erzeugten Daten finden sich diverse Umgebungsmerkmale korrekt erfasst wieder. So werden neben Wänden mit geradem oder geschwungenem Verlauf sowie Stützpfeilern auch kleinere geometrische Elemente korrekt aufgenommen. Raumelemente mit ausgeprägten Winkeln, beispielsweise Kanten zweier aufeinandertreffender Wände oder Ecken, werden weniger präzise erfasst und spiegeln sich in den Umgebungsdaten abgerundet wider.

Hingegen werden Fenster und andere transparente Flächen nicht erkannt und erscheinen im Polygonnetz, wie auch im letztlich erzeugten Plan, als Löcher. Hinter solchen Flächen zeichnen sich durch Reflektionen zudem häufig nicht der Realität entsprechende geometrische Elemente ab. Am Beispiel eines Fensters sind diese Eigenheiten des Erfassungsvorgangs in Abbildung 20 dargestellt. Generell werden Objekte mit reflektierenden Flächen, beispielsweise glänzend lackierte Möbel, gerade bei schwarzer oder dunkler Farbe, von der HoloLens 2 regelmäßig falsch wahrgenommen, wie in Abbildung 21 zu erkennen.

Abbildung 20: An Fenstern erzeugte Löcher im Polygonnetz sowie dahinterliegende Erfassungsfehler durch Reflektionen.

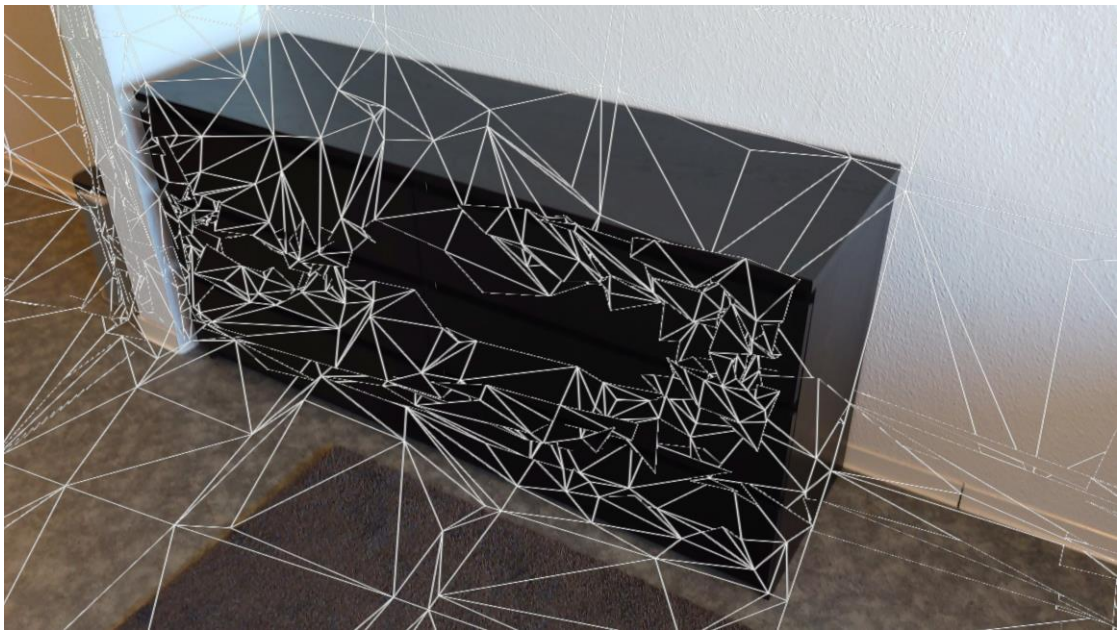

Abbildung 21: Aufgrund von Farbe und Oberflächenbeschaffenheit inkorrekte Erfassung eines Möbelstücks.

Auch bei größeren erfassten Bereichen wird das erzeugte Polygonnetz korrekt in der Umgebung positioniert. Es kommt also nicht zu Abweichungen zwischen weit entfernten Punkten, welche inkorrekte Krümmungen oder Übergänge des Netzes hervorrufen könnten. Dabei funktioniert die Erkennung beziehungsweise Wiedererkennung von Umgebungsbereichen zur Positionierung des Polygonnetzes gut und auch bei verschiedenen Lichtverhältnissen, solange diese ziemlich konstant sind. Auch weniger markante Umgebungsbereiche werden so gut erkannt und dem entsprechenden Abschnitt des Polygonnetzes zugeordnet.

Verdeckte Umgebungsmerkmale, beispielsweise solche, die sich hinter offenen Türen befanden, werden dabei nur bei expliziter Betrachtung durch den Vermessenden miterfasst.

6.2 Vollständigkeit der erzeugten Grundrisspläne

Der erzeugte Plan stellt eine insgesamt akkurate Abbildung des erfassten Raumaufbaus dar. In klassischen Grundrissplänen als einzelne Linien oder Kurven dargestellte Wandabschnitte werden dabei stattdessen durch viele kleine aneinandergereihte Linien dargestellt. So erscheinen eigentlich glatte Flächen im unbearbeiteten Plan eher unruhig.

Raummerkmale, beispielsweise Türen oder Fenster werden, anders als in gängigen Grundrissplänen, nicht gekennzeichnet, können jedoch erahnt werden, wie in Abbildung 22 zu sehen. Gerade Türen können dabei je nach Öffnungsgrad jedoch auch als Wandbestandteile erscheinen. Eine Herausstellung dieser Merkmale ist im Nachhinein händisch vorzunehmen.

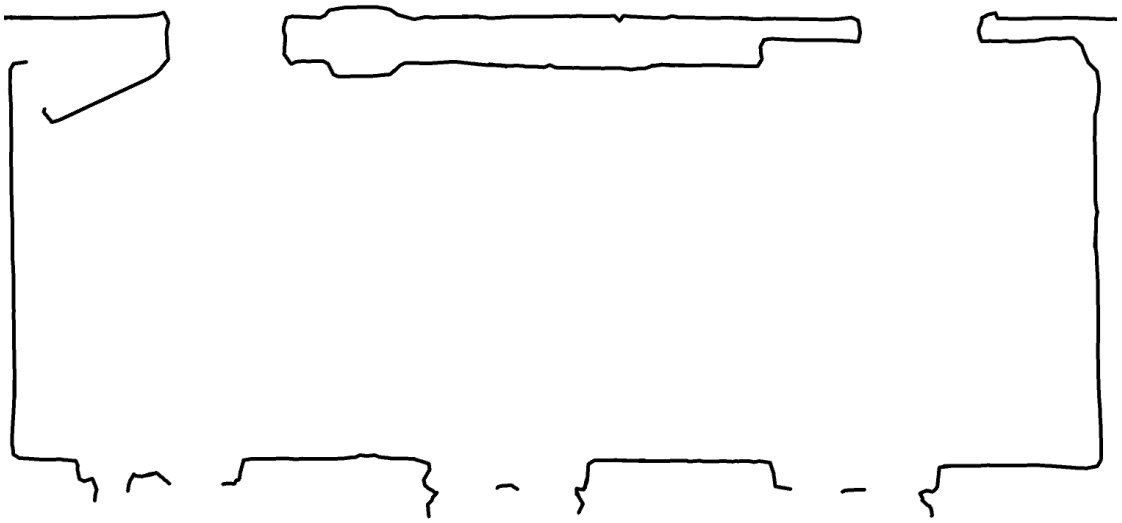

Abbildung 22: Darstellung einer geöffneten Tür (oben links) sowie einer Reihe an Fenstern (unten) in einem automatisiert erzeugten Plan.

Da der Plan auf einem Querschnitt in einer festgelegten Höhe basiert, werden geometrische Merkmale in anderen Höhen nicht im Grundrissplan erfasst. Gerade Dachschrägen werden daher nicht korrekt dargestellt und erscheinen als Wände an der gewählten Schnittposition. Eine händische Eintragung ihrer Position ist also notwendig.

Eine Bemaßung des Plans mit Wandlängen, -stärken und ähnlichen Elementeigenschaften geschieht ebenfalls nicht. Nach Erstellung des Plans kann diese allerdings durch Messungen am Plan selbst ermittelt und angefügt werden.

Weiterhin wird keine Kennzeichnung von Wandarten, innerhalb von Wänden befindlicher Schächte und ähnlicher Merkmale vorgenommen und ist, wie auch bei anderen Vermessungssystemen, manuell vorzunehmen.

6.3 Genauigkeit der Grundrisse

Ein augenscheinlicher Vergleich von mithilfe konventioneller Methoden erstellter Grundrisspläne mit solchen, welche anhand des in dieser Arbeit dargestellten Prozesses erzeugt wurden, lässt insgesamt eine deutliche Übereinstimmung erkennen. Anhand der Überlagerung entsprechender den gleichen Raumkomplexen zugehöriger Pläne, wie beispielhaft in Abbildung 23 dargestellt, ist zudem auffällig, dass erkennbare Abweichungen der Grundrisse voneinander eher abschnittsweise auftreten.

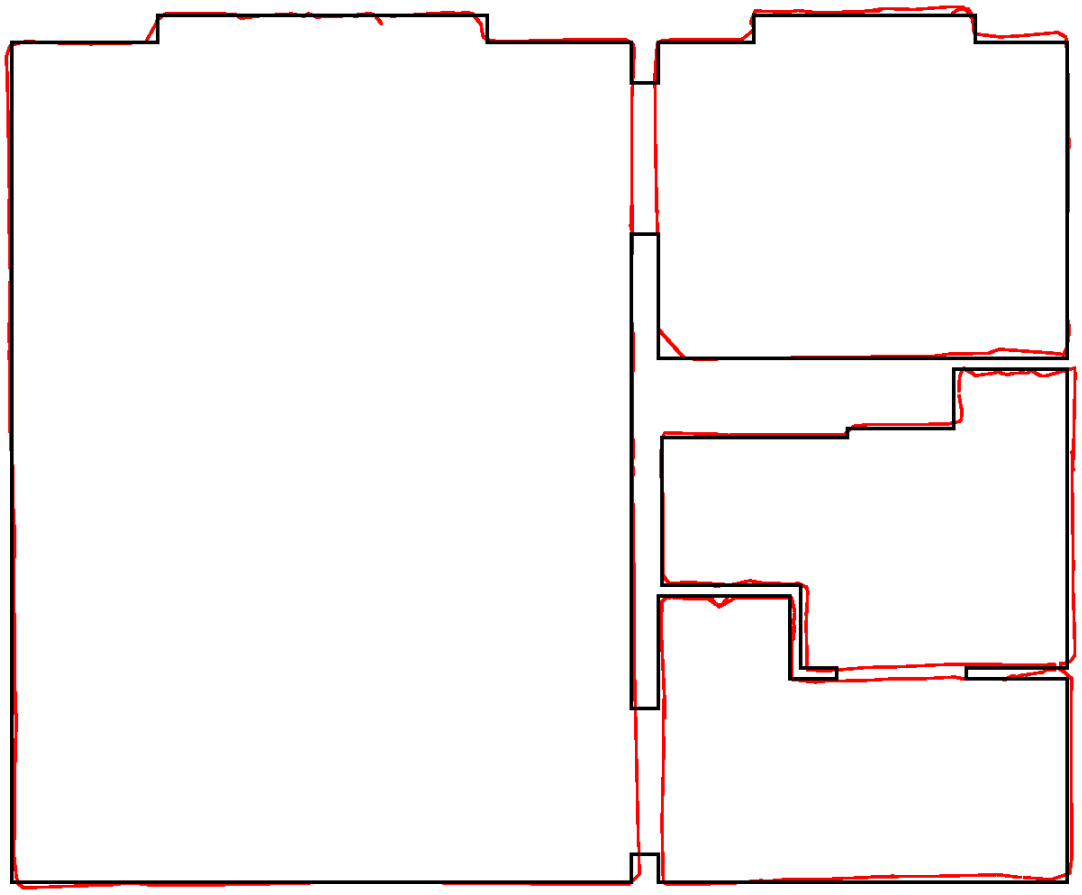

Abbildung 23: Vergleich eines durch Handaufmaß erstellten Grundrisses (schwarz) mit einem automatisiert Erzeugten (rot). Aufgrund der möblierungsbedingt gewählten Querschnittshöhe fehlen die Durchgänge im automatisiert erstellten Plan.

Für die konkrete Auswertung der erzielten Ergebnisse, in Bezug auf die im Plan dargestellten Maße, erstellte ich Grundrisspläne verschiedener Räumlichkeiten durch Handvermessung und über den auf der HoloLens 2 basierenden Prozess. Beispielhaft sollen im Folgenden die in Abbildung 24 und Abbildung 25 dargestellten bemaßten Pläne zur Visualisierung dieses Vergleichs von Wandlängen sowie -stärken und Raumflächen dienen. Der kleinste in den Plänen widergespiegelte Längenunterschied beträgt dabei 0,5 Zentimeter, da genauere Maßangaben in beiden Verfahren nicht zuverlässig möglich sind.

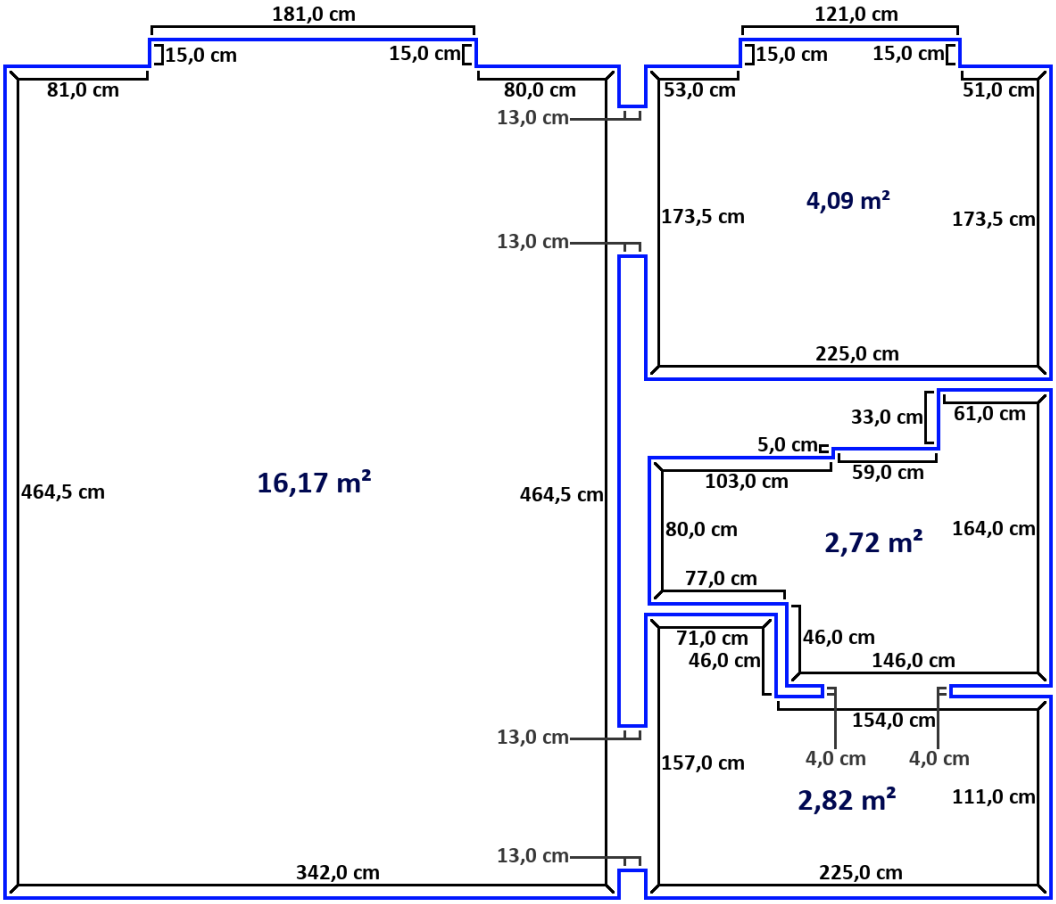

Abbildung 24: Durch Handvermessung erstellter Grundrissplan mit Bemaßung.

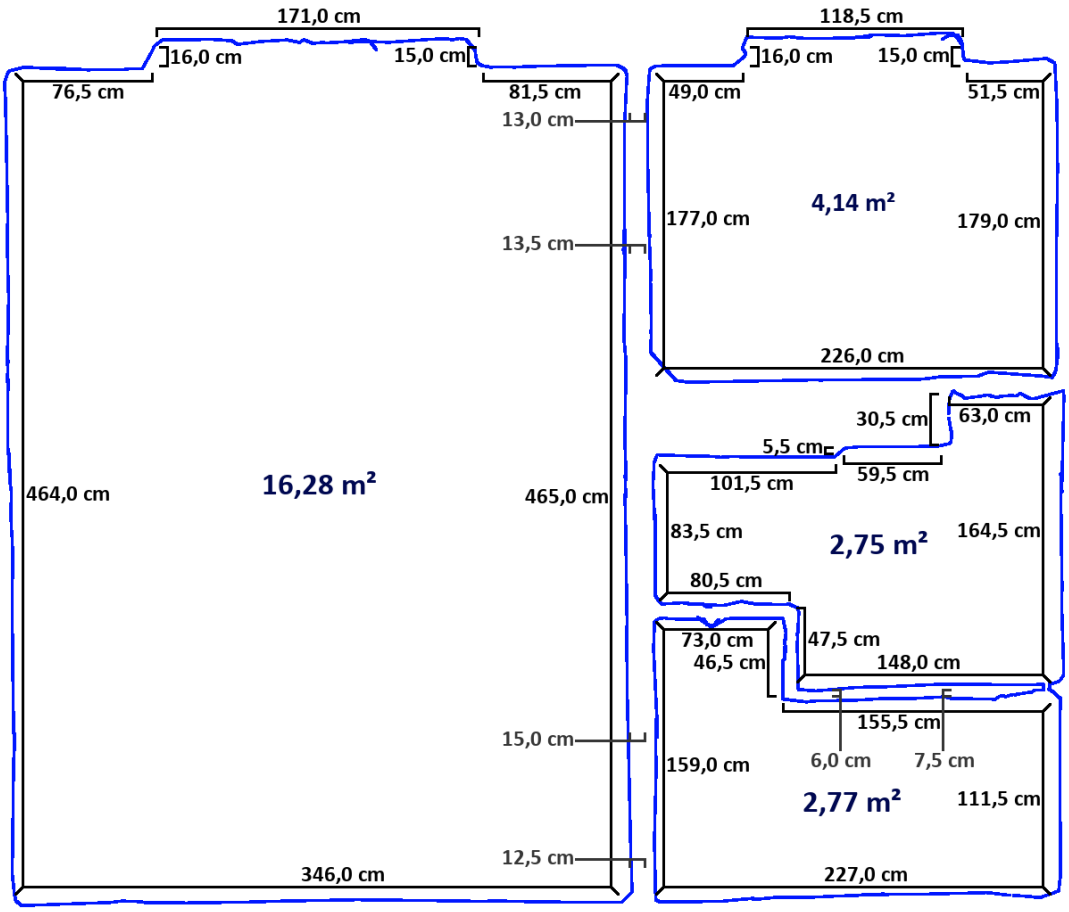

Abbildung 25: Automatisiert erstellter Grundrissplan mit händisch hinzugefügter Bemaßung.

Das Maßnehmen an den automatisiert erstellten Grundrissplänen wurde dadurch erschwert, dass Ecken darin zumeist abgerundet dargestellt werden, wodurch ihre genaue Position nicht klar erkennbar ist. Ich wählte bei der Bemaßung jeweils den Punkt als Eckpunkt, an welchem die aufeinandertreffenden Wände einander bei Verlängerung schneiden würden.

Die ermittelten Maße gleichen sich teilweise, in den meisten Fällen treten jedoch Abweichungen nach oben oder unten auf. Diese Abweichungen bewegten sich üblicherweise im Bereich von unter 5 Zentimetern und erreichten bei den angestellten Vergleichen einen Höchstwert von 10 Zentimetern.

Bei den ermittelten Wandstärken treten große Maßunterschiede auf, da bereits kleinere Abweichungen, wie sie auch bei den Wandlängen auftreten, aufgrund der geringen Dicke der Wände einen starken Einfluss auf das Messergebnis haben. Die Wandstärken variieren dabei stark und sind über die Länge der jeweiligen Wand hinweg nicht konstant, mit Abweichungen über oder unter dem tatsächlichen Wert.

Für ein akkurateres Ergebnis würde es sich bei gleichbleibend dicken Wänden dabei anbieten, aus mehreren Messergebnissen an einer Wand einen Mittelwert zu bestimmen. Durch

Verwendung eines solchen Mittelwertes als angenommene Wandstärke würden aufgrund der für die Messung gewählten Position stark von der Realität abweichende Ergebnisse verhindert, gleichzeitig aber dennoch ein ausreichend genaues Messergebnis erreicht werden.

Beim Vergleich der Raumflächen ergeben sich nur sehr geringe Abweichungen, wobei erneut kein klarer Trend zur größeren oder kleineren Abbildung der Flächen erkennbar ist. Da die Schwankungen durch zu große und zu kleine Messwerte sich größtenteils ausgleichen, hat die Raumgröße kaum Einfluss auf die entstehenden Abweichungen, bei größeren Räumen sinkt ihre Relevanz also eher. Gerade bei Handvermessung können dabei ähnlich große Schwankungen allein aufgrund unzureichend genauer Messungen entstehen.

Insgesamt sind die ermittelten Abweichungen für den betrachteten Entwicklungsstand akzeptabel. Gerade bei längeren Messstrecken beziehungsweise größeren zu vermessenden Flächen sind die auftretenden Ungenauigkeiten dabei wenig relevant. Kleinere Distanzen, wie sie gerade bei der Feststellung von Wandstärken von Bedeutung sind, werden jedoch deutlich von den auftretenden Abweichungen beeinflusst. Konkret bei Wandstärken kann diesem Problem jedoch durch Bildung eines Mittelwerts aus mehreren Messungen entgegengewirkt werden.

Ein sehr großes auftretendes Problem stellt die Unsicherheit bei der Feststellung der genauen Position von Ecken im Plan dar. Eine inakkurat festgelegte Position beeinflusst dabei die Längenermittlung bei den angrenzenden Wänden, das Ermitteln der Eckpunkte setzt jedoch eine Interpretation des Plans voraus und ist entsprechend inhärent ungenau.

Fehler bei der Datenerfassung geschehen nur selten in relevantem Umfang. Ihr Einfluss bei der Erstellung des Grundrisses kann zudem, durch Anpassung der Höhe der Querschnittsfläche, minimiert werden. Entsprechend stellten Fehler in den dem Plan zugrundeliegenden Daten bei den vorgenommenen Vergleichen kein Hindernis für die Bemaßung dar.

6.4 Arbeitsaufwand des Prozesses

Den zeitlich größten Aufwand benötigt im dargelegten Prozess der Schritt der Datenerfassung mit der HoloLens 2. Kleinere Räume können dabei in wenigen Minuten vermessen werden, auch größere Raumkomplexe können innerhalb einiger Stunden erfasst werden. Neben der Gesamtgröße der Räumlichkeiten wirkt sich dabei vor allem die Komplexität ihres Innenraums sowie die Menge an abseits der festen Gebäudebestandteile vorhandener Objekte, beispielsweise Möbel, auf die benötigte Zeit aus. Konzentriert sich die Datenerfassung dabei auf die Betrachtung von Wänden, im Optimalfall lediglich als Streifen in einer festgelegten

Höhe, und übergeht Decken, Böden und nicht fest zum Gebäude gehörende Objekte, kann der Prozess deutlich beschleunigt werden.

Der Export der ermittelten Daten ist ebenfalls zeitaufwändig, benötigt jedoch keine aktive Bearbeitung. Bei dem in diesem Projekt erreichten Stand ist der Export dabei unzuverlässig und schlägt teilweise fehl, wodurch sich die benötigte Zeit entsprechend verlängert. Der Prozess beziehungsweise der nötige Aufwand ist entsprechend unvorhersehbar.

Das Herunterladen der ermittelten Daten vom Gerät gestaltet sich über das WDP einfach und ist wenig zeitaufwendig. Eine aktive Überwachung des Prozesses ist dabei erneut nicht notwendig.

Das Erstellen des Grundrissplans aus den so gewonnenen Daten gestaltet sich ebenfalls zeitlich effizient. Lediglich der Import des Polygonnetzes in Unity sowie die Analyse der Umgebungsdaten erfordern ein kurzes Abwarten. Ansonsten geschieht der Prozess in Echtzeit, der zeitliche Aufwand steigt also je nachdem, wie sehr die Einstellungen zum Erreichen des gewünschten Ergebnisses angepasst werden müssen. Die sofortige Umsetzung der Änderungen an der Planvorschau minimieren dabei die benötigte Zeit.

Der letztliche Export des so erstellten Grundrissplans geschieht ohne Verzögerung und erzeugt für die direkte Weiterverarbeitung geeignete Dateien. Um einen vollwertigen Grundrissplan zu erhalten sind diese Dateien dabei nachzubearbeiten, wodurch weiterer Zeitaufwand einzuplanen ist.

7 Fazit und Zukunftsaussicht

Insgesamt kann festgehalten werden, dass es möglich ist, anhand der von der HoloLens 2 bereitgestellten Umgebungsdaten für einen Innenraum automatisiert einen schlichten Grundrissplan zu erstellen. Die Qualität beziehungsweise der Aussagegehalt des Plans ist dabei ausreichend, um den Aufbau des betrachteten Innenraumes abzubilden und anhand der Abbildung grobe Maße seiner Elemente zu ermitteln.

Für eine direkte praktische Verwendung sind die aktuell erzeugten Grundrisspläne nicht geeignet und eine Vereinfachung und/oder Erweiterung ihrer Inhalte ist notwendig. Sie stellen somit zunächst eine Grundlage für ausführlichere Pläne dar.

Der Prozess zur Planerstellung ist dabei insgesamt, gerade im Vergleich zu etablierten teilautomatisierten Verfahren, unkompliziert und mit entsprechender Anleitung auch von Laien durchführbar. Aus meiner Sicht steht der Verwendung der betrachteten Technologie für Vermessungszwecke nichts im Weg, einige innerhalb dieses Projekts zum aktuellen Stand auftretende Probleme verkomplizieren den Prozess jedoch.

So ist gerade die Verlässlichkeit des Datenexports von der HoloLens 2, aufgrund von häufigen Fehlern und Abbrüchen ohne entsprechende Hinweise, unzureichend und stellt das wohl wichtigste zu verbessernde Kriterium des ermittelten Ablaufs dar. Auch ist der begrenzte Einfluss Vermessender auf die Umgebungserfassung mit dem Gerät problematisch. Aktuell kann lediglich durch Anpassung von Betrachtungswinkel und -intensität auf Einflussnahme gehofft werden, ein direktes Steuern des Prozesses ist jedoch nicht möglich.

Die Weiterentwicklung der HoloLens-Gerätefamilie sollte die innerhalb des Projekts genutzten Funktionen optimieren beziehungsweise erweitern und den Prozess, gerade im Hinblick auf vollständige und verlässliche Datenerfassung sowie Erfassungsgenauigkeit, verbessern. So würde eine umfassendere Nutzung zur Vermessung ermöglicht werden.

Auf Basis des aktuell umgesetzten Projekts ist aber auch eine Weiterentwicklung des behandelten Prozesses möglich. So bietet sich, wie in Kapitel 5.2.3 erläutert, die Möglichkeit an, den erzeugten Querschnitt automatisiert zu vereinfachen. Auch kann der Prozess der Datenanalyse und Querschnittserstellung, wie ebenfalls in diesem Kapitel erläutert, optimiert werden, um gerade bei größeren Innenräumen keine Leistungsprobleme in der Anwendung hervorzurufen.

Zuletzt bietet sich eine Umsetzung als Unity-unabhängiges Programm an. Der erreichte Stand setzt eine Installation der korrekten Unity-Version sowie Kenntnis von Unity selbst voraus, wobei ein Großteil der Funktionen des Unity-Editors für Personen, welche die Anwendung

nutzen, nicht relevant sind. Das eigentlich schlichte Werkzeug zur Querschnittserstellung wird so überfrachtet und auch seine Leistung sinkt aufgrund der zusätzlich im Hintergrund laufenden Prozesse des Editors. Auch ist die Anwendung aktuell sehr unübersichtlich und sollte für eine tatsächliche Nutzung angenehmer gestaltet werden. Die weitere Nutzung von Unity würde sich dazu anbieten, da es die Erstellung alleinstehender Anwendungen ermöglicht und so die bereits erstellten Skripte und Abläufe ohne starke Anpassungen weiterverwendet werden könnten. Konkret wäre dabei zunächst die Umsetzung einer grafischen Oberfläche zur grundlegenden Prozessinteraktion in der Spielansicht sowie die Integration einer Anleitung zur Verwendung der Anwendung vorrangig.

Insgesamt wurde die Zielstellung des Projekts dennoch erfolgreich erreicht und ein Prozess mit Potential für zukünftigen Ausbau geschaffen.

V Literaturverzeichnis

[1] Autodesk, Inc., „CAD Software - Computergestütztes Design“, Autodesk, Inc., [Online]. Verfügbar unter: https://www.autodesk.de/solutions/cad-software. [Zugriff am 19 Januar 2021].

[2] B. Bray, „What is Mixed Reality? “, Microsoft Corporation, 26 August 2020. [Online]. Verfügbar unter: https://docs.microsoft.com/en-us/windows/mixed-reality/discover/mixed-reality. [Zugriff am 19 Januar 2021].

[3] 1. I. SE, „Pixelgrafik oder Vektorgrafik? Vor- und Nachteile im Vergleich“, 1&1 Ionos SE, 28 Oktober 2020. [Online]. Verfügbar unter: https://www.ionos.de/digitalguide/websites/webdesign/pixelgrafik-vs-vektorgrafik-ein-vergleich/. [Zugriff am 7 Februar 2021].

[4] W. Freeden und R. Rummel, Ingenieurgeodäsie: Handbuch der Geodäsie, herausgegeben von Willi Freeden und Reiner Rummel, 1 Hrsg., Berlin: Springer-Verlag GmbH, 2017, pp. 23-45.

[5] M. Hugenschmidt, Lasermesstechnik: Diagnostik der Kurzzeitphysik, Berlin, Heidelberg: Springer-Verlag GmbH, 2007, pp. 109-122.

[6] Spektrum Akademischer Verlag, „Lidar“, Spektrum Akademischer Verlag, 1999. [Online]. Verfügbar unter: https://www.spektrum.de/lexikon/optik/lidar/1875. [Zugriff am 2 Januar 2021].

[7] Spektrum Akademischer Verlag, „Photogrammetrie“, Spektrum Akademischer Verlag, 2000. [Online]. Verfügbar unter: https://www.spektrum.de/lexikon/geowissenschaften/photogrammetrie/12220. [Zugriff am 2 Januar 2021].

[8] A. Tuliper, „Introduction to the HoloLens“, *MSDN Magazine,* pp. 48-52, November 2016.

[9] Microsoft Corporation, „HoloLens 2: Technische Daten und Funktionen“, Microsoft Corporation, [Online]. Verfügbar unter: https://www.microsoft.com/de-de/p/HoloLens-2/91pnzzznzwcp. [Zugriff am 17 Januar 2021].

[10] J. Langston, „New HoloLens 2 gives Microsoft the edge in the next generation of computing“, Microsoft Corporation, 24 Februar 2019. [Online]. Verfügbar unter: https://news.microsoft.com/innovation-stories/hololens-2/. [Zugriff am 16 Januar 2021].

[11] P. Meisel, „Microsoft HoloLens 2: Microsoft zeigt die Zukunft von Mixed Reality“, Microsoft Corporation, 26 Februar 2019. [Online]. Verfügbar unter: https://news.microsoft.com/de-de/microsoft-hololens-2/. [Zugriff am 8 Januar 2021].

[12] A. Tuliper, „Introduction to the HoloLens, Part 2: Spatial Mapping“, *MSDN Magazine,* pp. 26-31, Januar 2017.

[13] M. Zeller, „Spatial mapping“, Microsoft, 21 März 2018. [Online]. Verfügbar unter: https://docs.microsoft.com/en-us/windows/mixed-reality/design/spatial-mapping. [Zugriff am 10 Januar 2021].

[14] Microsoft Corporation, „Scene understanding“, Microsoft Corporation, 8 Juli 2019. [Online]. Verfügbar unter: https://docs.microsoft.com/en-us/windows/mixed-reality/design/scene-understanding. [Zugriff am 10 Januar 2021].

[15] Microsoft Corporation, „Scene understanding SDK overview“, Microsoft Corporation, 14 Dezember 2020. [Online]. Verfügbar unter: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/platform-capabilities-and-apis/scene-understanding-SDK. [Zugriff am 11 Januar 2021].

[16] Microsoft Corporation, „HoloLens environment considerations“, Microsoft Corporation, 29 August 2019. [Online]. Verfügbar unter: https://docs.microsoft.com/en-us/hololens/hololens-environment-considerations. [Zugriff am 11 Januar 2021].

[17] Microsoft Corporation, „Einführung in die Mixed Reality-Entwicklung“, Microsoft Corporation, [Online]. Verfügbar unter: https://docs.microsoft.com/de-de/windows/mixed-reality/develop/development. [Zugriff am 16 Januar 2021].

[18] J. McCulloch und M. Wilter, „4. Positioning objects in the scene“, Microsoft Corporation, 01 Juli 2020. [Online]. Verfügbar unter: https://docs.microsoft.com/en-us/windows/mixed-reality/develop/unity/tutorials/mr-learning-base-04. [Zugriff am 31 Januar 2021].

[19] Free Software Foundation, Inc., „GNU Lesser General Public License“, Free Software Foundation, Inc., 29 Juni 2007. [Online]. Verfügbar unter: https://www.gnu.org/licenses/lgpl-3.0.de.html. [Zugriff am 31 Januar 2021].