Visualisierungsinstitut der Universität Stuttgart (VISUS)

Universität Stuttgart
Allmandring 19
D–70569 Stuttgart

Bachelorarbeit

Virtuelle Vervollständigung einer Ruine mithilfe von Augmented Reality

Lukas Ganter

Studiengang:	Softwaretechnik
Prüfer/in:	Prof. Dr. Michael Sedlmair
Betreuer/in:	Nina Dörr, M.Sc., Michael Becher, M.Sc.
Beginn am:	17. November 2022
Beendet am:	17. Mai 2023

Kurzfassung

Durch Alterungsprozesse oder menschengemachte Zerstörung verkommen viele wichtige und interessante Orte der Geschichte zunehmend zu Ruinen. Moderne Technologien bieten dabei Möglichkeiten, diese Orte auf verschiedene Art und Weise virtuell zu rekonstruieren.
Diese Arbeit beschäftigt sich mit der virtuellen Rekonstruktion von Ruinen mithilfe einer Augmented Reality (AR) Anwendung. Das Ziel ist eine Darstellung des originalen Zustandes eines Gebäudes, ohne vorhandene Elemente der Ruine zu überdecken. Hierfür werden in dieser Arbeit verschiedene Technologien eingesetzt. Diese umfassen einen Laserscanner für das virtuelle Abbild der Ruine, Software zum Erstellen von 3D Grafiken, eine Spieleengine zum Aufbau des virtuellen Teils der Rekonstruktion und die AR Brille HoloLens 2, mit welcher die echten und die virtuellen Teile der Arbeit verbunden werden.
Zusätzlich werden andere Arbeiten dieses Themenfeldes betrachtet und in verschiedenen Gesichtspunkten mit dem Ergebnis dieser Arbeit verglichen. Die Rekonstruktion mit dieser Technik bietet eine neue Möglichkeit, dem Benutzer originale Gegebenheiten vor Ort darzustellen. Das exakte Kombinieren der realen und virtuellen Komponenten stellt eine der größten Herausforderungen dar. Für aktuell nicht gelöste Probleme werden mögliche weiterführende Verbesserungen aufgezeigt und diskutiert. Außerdem werden Möglichkeiten vorgestellt, welche den Informationsgehalt der Rekonstruktion noch weiter verbessern können.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Die Leistungsfähigkeit von immer kompakter werdenden mobilen Computern wächst seit einigen Jahren an. Davon profitieren unter anderem die Bereiche der Virtual Reality (VR) und Augmented Reality (AR).
Das Ziel von VR ist die Generierung von virtuellen Welten, die dann dem Benutzer auf einem Display dargestellt werden können. Die VR Technologie hat viele verschiedene Anwendungsgebiete, darunter die Industrie 4.0 und die Unterhaltungsindustrie.
AR stellt einen Mittelweg zwischen der Realität und VR dar, indem sie reale und virtuelle Elemente kombiniert. Anwendungsgebiete für diese Technologie sind beispielsweise lokal begrenzte Hologramme oder Navigationshilfen wie Pfeile. Diese virtuellen Elemente können dann wie bei einem Head-Up Display auf eine Glasfläche vor die Augen des Nutzers projiziert werden.
Anwendungsgebiete für diese Technologie sind vielfältig. Neben der Unterhaltungsindustrie gibt es viele weitere Forschungsgebiete, etwa für die Industrie, den Tourismus oder die Forschung. Vor allem im Tourismussektor spielt AR heutzutage kommerziell noch keine nennenswerte Rolle. Mit Verbesserungen in der Leistungsfähigkeit und sinkenden Anschaffungskosten durch Massenproduktion hat diese Technologie in den nächsten Jahren aber das Potential den Sektor zu revolutionieren.
Diese Arbeit beschäftigt sich mit der virtuellen Rekonstruktion von Ruinen mithilfe von AR. Dabei liegt der Fokus auf dem Erhalten des Bildes der vorhandenen Ruinen und dem virtuellen Rekonstruieren fehlender Teile. Zusätzlich sollen virtuell Informationen zur Ruine und zu speziellen Teilbereichen wie Bautechniken gegeben werden. Die Arbeit zeigt, dass die virtuelle Rekonstruktion eine gute Übersicht über die originalen Gegebenheiten bieten kann. Allerdings werden auch Einschränkungen deutlich, so zum Beispiel bei der exakten Kombination der realen und virtuellen Teile. Zunächst werden in Kapitel 2 die grundlegenden Technologien und Orte eingeführt, mit welchen diese Arbeit realisiert wird. Im Anschluss behandelt Kapitel 3 Arbeiten mit ähnlichen Zielen und vergleicht sie in verschiedenen Gesichtspunkten mit diesem Projekt. Die eigentliche Umsetzung der Arbeit wird in Kapitel 4 beschrieben. Dieses Kapitel umfasst die Entwicklung eines Mockups sowie die eigentliche Rekonstruktion der gewählten Ruine. Daran anschließend wird in Kapitel 5 das Ergebnis vorgestellt. Außerdem werden Limitierungen und mögliche Erweiterungen diskutiert sowie die weitere Verwendbarkeit eingeordnet. Zum Schluss werden in Kapitel 6 die Ergebnisse der Arbeit zusammengefasst.

2 Grundlagen

Dieses Kapitel beschäftigt sich mit den Grundlagen dieser Arbeit. Dazu zählen die wichtigsten Teile der verwendeten Hardware und Software. Außerdem wird die Ruine vorgestellt, welche im Zuge dieser Arbeit virtuell rekonstruiert werden soll.

2.1 Hardware

Das erste Kapitel führt in die für die Arbeit verwendete Hardware ein. Dazu zählen ein Laserscanner zum Scannen der vorhandenen Ruinenteile, sowie die HoloLens 2, eine AR Brille des Herstellers Microsoft [Mic23].

2.1.1 Laserscanner

Der für die Arbeit verwendete Laserscanner ist der Leica BLK360[Lei23]. Dieser erstellt ein extrem genaues dreidimensionales Abbild der Umgebung im Umkreis von 360° um den Scanner. Als Ergebnis erhält man eine Punktwolke im Raum, die im Anschluss mithilfe anderer Tools weiter bearbeitet werden kann. In dieser Arbeit wird mit dem Scanner die Ruine des römischen Hauses eingescannt. Die entstehende digitale Repräsentation ist für die weitere Rekonstruktion der fehlenden Gebäudeteile notwendig.

2.1.2 HoloLens 2

Die HoloLens 2 ist ein Head Mounted Display (HMD) von Microsoft [Mic23], mit welchem AR Anwendungen dargestellt werden können. Sie funktioniert, indem auf Gläser, welche sich vor den Augen des Benutzers befinden, der virtuelle Teil der Anwendung projiziert wird. Während der Benutzer sich bewegt, erstellt die Brille automatisch eine dreidimensionale Karte der realen Umgebung. Diese Geometrie wird Spatial Mesh genannt. Weiterhin kann die Brille automatisch ihre Position im Raum erkennen und diese kontinuierlich aktualisieren. Das Koordinatensystem, in welchem das Spatial Mesh definiert ist, wird als Spatial Coordinate System bezeichnet. Es gibt verschiedene Möglichkeiten für den Benutzerinput, wie beispielsweise Handgesten. Dies erlaubt dem Benutzer in Kombination mit einem angepassten dreidimensionalen Benutzerinterface die Bedienung einer Anwendung durch Handbewegungen im Raum. Für dieses Projekt ist sowohl die Erkennung der realen Position als auch die Steuerung über Handbewegungen relevant. Weitere Funktionalitäten der Brille schließen unter anderem eine Spracherkennung und Blickrichtungserkennung ein.

2.2 Software

Dieser Teil führt die wichtigste verwendete Software ein. Diese umfassen die Spieleengine Unreal Engine 5 [Epi23b] für die Anwendungserstellung sowie das Grafikprogramm Blender [Ble23] zur Erstellung der benötigten virtuellen Objekte.

2.2.1 Unreal Engine 5

Die Unreal Engine 5 (UE5) [Epi23b] ist eine Spieleengine von Epic Games [Epi23a]. Im Gegensatz zu Engines wie Unity [Uni23] stellt UE5 ein Paket dar, in welches bereits viele für die Spieleentwicklung wichtigen Komponenten eingebunden sind. Dies sorgt für eine solide Basis für viele Projekte, auch weil die einzelnen Teile perfekt aufeinander abgestimmt sind. Trotzdem kann je nach Anforderung der Funktionsumfang der Engine mithilfe von Plugins erweitert werden. Dafür stehen im offiziellen Unreal Marketplace [1] viele kostenlose und kostenpflichtige Erweiterungen zum Download bereit. Zusätzlich gibt es auch die Möglichkeit, unabhängig vom Marketplace Erweiterungen von Drittanbietern zu installieren. Für dieses Projekt sind einige Plugins erforderlich, um die begrenzten Funktionen der Engine für die AR Entwicklung zu erweitern.
Das erste Plugin nennt sich Microsoft OpenXR [Mic22a] und ist eine offizielle Erweiterung von Microsoft. Diese integriert das offizielle Framework OpenXR von Khronos [Khr23] in die Engine. Dieser offene und lizenzfreie API-Standard bietet die Möglichkeit, mit UE5 ein Projekt zu entwickeln, welches dann ohne große Anpassungen auf verschiedenen AR Geräten installiert werden kann. Zwei der kompatiblen Geräte sind die Varjo XR-3 [Var23] oder die in dieser Arbeit verwendete HoloLens 2 (vgl. Abschnitt 2.1.2).
Für weitere Funktionalitäten des Benutzerinputs werden auch die Plugins Mixed Reality UX Tools und Mixed Reality UX Tools Examples [Mic22b] verwendet. Diese sind ebenfalls offizielle Erweiterungen von Microsoft und bringen verschiedene Funktionalitäten mit, wie beispielsweise 3D Buttons und andere UI Elemente.
Weiterhin wird auch die Erweiterung World Locking Tools [Mic22c] von Microsoft eingesetzt. Diese bietet Funktionalitäten, um mit sogenannten Weltankern die reale und virtuelle Welt zu verbinden. Weltanker sind Objekte, die fest in der virtuellen Szene platziert sind. Um diesen Objekten eine Position und Ausrichtung im realen Raum zuzuweisen, stehen verschiedene Optionen zur Auswahl. Die manuelle Platzierung ist sehr einfach umzusetzen, allerdings fehlt hier unter Umständen die benötigte Präzision. Außerdem ist eine automatische Korrektur der Platzierung nicht möglich.
Für dieses Projekt wurde deshalb die zweite Möglichkeit gewählt, die Platzierung mithilfe von QR-Codes. Diese können ausgedruckt an die gewünschten Stellen gelegt werden. Wenn die Codes dauerhaft in der realen Welt verbleiben, kann die Brille die Position der Weltanker kontinuierlich neu einlesen. Das führt zu einer akkurateren Platzierung, besonders wenn der Benutzer sich bewegt. Wenn die reale Position der Weltanker einmal gesetzt ist, hält die Software der HoloLens 2 diese an der gesetzten Position. Ein Update der Platzierung zur Laufzeit ist nicht nötig, kann aber, wie zuvor erwähnt, die Präzision erhöhen.
Dieses Projekt verwendet zum Erstellen aller nötiger Funktionen ausschließlich das visuelle Scripting System, genannt Blueprints, welches in der Unreal Engine standardmäßig enthalten ist.

[1] Unreal Marketplace: https://www.unrealengine.com/marketplace/

Die Programmiersprache C++, die ebenfalls verwendet werden kann, kommt nicht zum Einsatz. In Abschnitt 4.1.2 wird eine mögliche Anwendung von C++ beschrieben, die aber nicht in das Projekt übernommen wurde.

2.2.2 Blender

Für die Erstellung der virtuellen Modelle wird das Open Source Programm Blender [Ble23] verwendet. Blender ist eine Software zum Erstellen von dreidimensionalen Objekten. Der große Funktionsumfang erlaubt hierbei eine hohe Flexibilität bei der Erstellung. Für dieses Projekt werden die Modelle teilweise anhand von Datenimporten angefertigt, während andere von Grund auf selbst modeliert werden. Näheres zu der Verwendung von Blender in dieser Arbeit ist in Abschnitt 4.2.2 beschrieben. Der Export der Modelle erfolgt im FBX Format. Dieser Dateityp kann direkt in die Unreal Engine importiert werden.

2.3 Römisches Haus

Als Grundlage für die Rekonstruktion dient in dieser Arbeit das römische Haus im Stuttgarter Rotwildpark [Pla05]. Es wurde im 2. Jahrhundert nach Christus gebaut. Zum jetzigen Zeitpunkt stehen nur noch die restaurierte Reste der Grundmauern (siehe Abbildung 2.1). Mehrmalige Ausgrabungen im 20. Jahrhundert lieferten wenig Hinweise auf die Art des Gebäudes, eine Nutzung als Wohnhaus konnte vor allem aufgrund des Grundrisses und der abgelegenen Lage aber weitgehend ausgeschlossen werden. Inzwischen wird vermutet, dass es sich um eine rituelle Kultstätte der damaligen Bevölkerung gehandelt hat.
Die in dieser Arbeit erstellte Vervollständigung des Gebäudes basiert auf einem Rekonstruktionsversuch der Archäologen, unter anderem zu sehen in diesem Artikel [Pla05]. Dieser wiederum stützt sich auf die vorhandenen Mauerreste sowie auf Fundamente, welche im Boden gefunden wurden, heutzutage aber nicht mehr sichtbar sind.
Dieses Gebäude wird aufgrund der einfachen Geometrie und der günstigen geographischen Lage für diese Arbeit verwendet.

Abbildung 2.1: Übersicht über die vorhandenen Reste des römischen Hauses in Stuttgart (Quelle: Hansjörg Lipp, http://geo.hlipp.de/photo/18064, Lizenz: Attribution-ShareAlike 2.0 Generic (https://creativecommons.org/licenses/by-sa/2.0/))

3 Verwandte Arbeiten

Bereits lange vor der Einführung der ersten Computer haben sich Menschen mit der Visualisierung von Umgebungen beschäftigt. Im Jahr 1788 erfand der Ire Robert Barker mit dem erst später so benannten Panoramatheater eine der ersten Interpretationen einer virtuellen Realität [Hei]. Diese Ausstellung bestand aus einem zylindrischen Ausstellungsraum, dessen Innenwände mit einem Landschaftspanorama bemalt waren. Besucher, die in der Mitte standen, konnten so eine 360° Ansicht der Landschaft bewundern. Diese Art der Präsentation wurde im Anschluss für viele weitere Ausstellungen verwendet.

Neben der eigentlichen Gebäuderekonstruktion ist auch die Informationsweitergabe über virtuelle Schautafeln ein wichtiger Teil dieser Arbeit. Durch weitergehende Informationen über die Ruine ist es für den Benutzer einfacher, sich vor Ort zurechtzufinden und bauliche oder örtliche Gegebenheiten im historischen Kontext zu verstehen. [BDPC12] beschäftigt sich ausführlich mit der Kombination von realen archäologischen Überresten und zusätzlichen virtuellen Informationen. Dabei liegt der Fokus nicht auf der Rekonstruktion der Ruinen, sondern auf dem Anzeigen der Position interessanter Stellen. Außerdem werden weitergehende Informationen zu diesen dargestellt. Bernardini et al. wählen das Mobiltelefon als Hardwareplattform. Das hat den Vorteil, dass prinzipiell jeder Besucher diese App installieren kann, und somit keine spezielle und teure zusätzliche Hardware benötigt wird. Ein Nachteil der Verwendung von Mobiltelefonen ist allerdings, dass der Benutzer die Augmented Reality Ansicht nur auf dem Display ansehen kann, also auch die reale Welt nur auf dem Display sieht. Das sorgt zumindest teilweise für einen Verlust der Verbindung zwischen realer und virtueller Welt. Für die Gebäuderekonstruktion ist es von Vorteil, optical-see-trough AR zu verwenden, wie es beispielsweise die HoloLens 2 bietet.
Ein weiteres Feature der zitierten Arbeit ist, dass jeder Benutzer für ihn oder sie interessante Stellen direkt in der App markieren kann. Dies erlaubt es Wissenschaftlern, die Datenbank der Anwendung kontinuierlich nach Benutzerwunsch zu erweitern.

Zusätzlich zu rein informellen Einblendungen kann das Smartphone auch benutzt werden, um graphische Rekonstruktionen darzustellen. Eine der einfachsten Möglichkeiten stellt hierbei das markerbasierte Anzeigen von Originalfotos dar. Um Besuchern eine tiefergehende Erfahrung zu ermöglichen, wurde dieses Verfahren für den abgerissenen Stadtteil Pimatgol in Seoul verwendet [KR10]. Dieses Projekt gibt dem Benutzer die Möglichkeit, an durch Marker festgelegten Orten Originalfotos des Bereichs vor dem Abriss zu betrachten. Wenn der Nutzer mit einem dieser Marker interagiert, wird das Fotos auf dem Bildschirm des Smartphones angezeigt.

Die in dieser Arbeit beschriebene Methodik zur Rekonstruktion der Ruine wird in vielen Bereichen eingesetzt. [SB11] beschreibt eine Anwendung zur Virtualisierung und anschließender Darstellung von Ruinenteilen und Artefakten. Ziel der zitierten Arbeit ist es, Artefakte in ihrem historischen Aussehen digital zu erhalten und reale Restaurationsarbeiten zu erleichtern. Hierzu werden zuerst die zu virtualisierenden Teile mithilfe eines Laserscanners eingescannt. Diese Daten werden dann zu einem dreidimensionalen Dreiecksmesh des Objektes verarbeitet, welches dann

mithilfe einer AR Brille angezeigt werden kann. Dabei werden verschiedene Modifikationen der Virtualisierung unterschieden. Diese Arbeit legt vor allem Wert auf die archäologische Korrektheit der Rekonstruktion, mit einem klaren Fokus auf technische Anwendungen zur realen Restaurierung von Objekten. Gleichzeitig können die Modelle aber auch für Visualisierungszwecke abseits der technischen Restaurierung eingesetzt werden. Diese Modelle können überall betrachtet werden und sind nicht ortsgebunden. Das ist ein klares Unterscheidungsmerkmal zu dieser Arbeit, welche für eine realistische Erfahrung den realen Ort der Ruine benötigt. Zwar ist es möglich, das Level überall zu öffnen und damit den virtuellen Teil der Rekonstruktion ortsungebunden zu betrachten. Informationstafeln mit Hinweisen zu realen Objekten vor Ort, der Richtungsweiser in der Mitte oder die passgenaue Kombination mit den realen Mauern verlieren dabei aber ihren darstellerischen Wert.

Eine Alternative zu der in dieser Arbeit verwendeten Unreal Engine stellt die Unity Game Engine dar [Uni23]. Diese bietet über verschiedene Plugins ähnliche Möglichkeiten zur AR Entwicklung. Gleichzeitig ist sie unter anderem für leistungsschwache Geräte optimiert, was sie für die Entwicklung von Anwendungen für ein Smartphone attraktiv macht. Eine solche Smartphoneanwendung stellt die Rekonstrukion der Kriegsruinen auf der Insel Corregidor dar [Ben15] [FDC+19]. Diese Anwendung ähnelt in vielen Punkten dem im Laufe dieser Arbeit erstellten Projekt. Es werden virtuelle Rekonstrutionen der Gebäude erstellt, welche auf dem Smartphonedisplay an der korrekten realen Position dargestellt werden. Der Benutzer kann sich überdies mit dem Handy bewegen und so Gebäude von allen Seiten betrachten. Der Unterschied zu dieser Arbeit, wie auch im nachfolgend beschriebenen Projekt Archeoguide [VKT+01], ist die Interaktion der Rekonstruktionen mit den realen Ruinen. In beiden zitierten Arbeiten werden die virtuellen Objekte über die vorhandenen Ruinen gelegt. Für diese Arbeit liegt der Fokus darauf, die vorhandenen Ruinenteile sichtbar zu lassen und nur fehlende Objekte virtuell zu ergänzen.

Eines der weiter fortgeschrittenen Projekte dieser Art ist Archeoguide [VKT+01]. Archeoguide verlässt sich im Gegensatz zu dieser Arbeit auf eine Mischung von GPS und Kompassdaten zum Tracken der Position des Nutzers. Dieses Tracking hat laut Vlahakis et al. eine Abweichung beim Tracking von „weniger als einem Meter“ ([VKT+01], S. 2). Unter normalen Umständen ist das Tracking der HoloLens 2 deutlich genauer. Dabei spielt die deutlich gestiegene Leistungsfähigkeit mobiler Rechengeräte eine wichtige Rolle. Zu Zeiten der Erstellung des Archeoguide Projektes war diese Art des komplexen Trackings auf mobilen Geräten noch nicht möglich. Die HoloLens 2 hat mit dem Qualcomm Snapdragon 850 eine aktuelle Recheneinheit mit acht Kernen und acht Threads integriert. Diese takten mit bis zu 2,96 GHz ([Qua]). Zusätzlich verfügt sie über 4GB Arbeitsspeicher ([Mic]). Dem gegenüber steht die verwendete Hardware für das Archeoguide Projekt. Als leistungsstärkste Clienteinheit ist hier ein Laptop mit 800MHz Prozessortaktung und 256MB angegeben [VKT+01]. Diese Leistungsdiskrepanz verdeutlicht die unterschiedlichen Möglichkeiten, die den Projekten technisch zur Verfügung stehen. Das von Archeoguide verwendete GPS Lokalisierungssystem in Kombination mit fest positionierten Referenzbeacons kann aber, vor allem für größere Ruinen, ein sinnvoller Erweiterungsansatz sein (siehe Abschnitt 5.5.4). Im Gegensatz zu der verwendeten Hardware im Projekt Archeoguide bietet die für dieses Projekt verwendete HoloLens einen deutlich erhöhten Tragekomfort, da keine externen Teile wie GPS-Empfänger oder ein Laptop im Rucksack mitgeführt werden müssen. Zusätzlich dazu bietet sie durch ihre gesteigerte Rechenleistung ein deutlich präziseres und angenehmeres Erlebnis. Letzteres vor allem durch die höhere Bildrate der Visualisierung. Ein weiteres Unterscheidungsmerkmal, wie im vorherigen Abschnitt erwähnt, ist die Art und Weise der Verknüpfung von realer Welt und virtueller

Vervollständigung. Archeoguide platziert, wie auch die Rekonstruktion von Corregidor [FDC+19], ein vollständiges virtuelles Gebäude am historisch korrekten Ort. Dabei werden vorhandene Ruinen überdeckt.

Die Erkennung der genauen Position und Ausrichtung des Benutzers im Bezug zum betrachteten Objekt ist äußerst rechenintensiv. Es gibt verschiedene Herangehensweisen zum Vereinfachen dieser Aufgabe. Das zuvor beschriebene Anzeigen von textbasierten Informationen oder originalen Bildern benötigt oftmals keine hohe Genauigkeit bei der Positionierung [BDPC12] [KR10]. Eine andere Möglichkeit ist es, die Position, von welcher ein Benutzer die Umgebung betrachtet, genau festzulegen. Dann muss zum Kombinieren der realen und virtuellen Ansichten nur noch die Orientierung berücksichtigt werden. Das wiederum sorgt dafür, dass keine komplexen Berechnungen zur Bilderkennung nötig sind, sondern lediglich Geometrische. Diese Technik wurde unter anderem für die Rekonstruktion des Gartens von Yuangmingyuan angewandt [HLW09]. Die dafür verwendete Hardware ähnelt den Teleskopen, welche für Touristen an vielen Orten aufgestellt zu finden sind. Das Bild wird in diesem Projekt jedoch rein virtuell auf zwei Displays übertragen, und enthält zusätzlich zu dem realen Kamerabild virtuelle Rekonstruktionen. Die virtuelle Darstellung ist dabei abhängig von der bekannten Position des Gerätes im Park und von der Ausrichtung, welche vom Nutzer frei gewählt werden kann und über Sensoren an die Recheneinheit übermittelt wird.

Eines der Hauptanwendungsgebiete für Software dieser Art ist der Tourismussektor. Viele Arbeiten haben sich bereits mit der Zukunft von AR Technologien für den Tourismus beschäftigt [Gut10] [LE21]. Die zuvor vorgestellte Arbeit über die Ruinen auf Corregidor [FDC+19] enthält eine Studie über die Anwendung. Diese zeigt, dass es die virtuelle Rekonstruktion vielen Anwendern erlaubt, tiefer in die Geschichte der Insel einzusteigen. Damit steigert sich auch die Qualität des Besuchs. Die Arbeit über die Rekonstruktion des Gartens von Yuangmingyuan geht bei der Benutzerstudie noch einen Schritt weiter. Hier wird neben dem Evaluation über den Nutzen der Anwendung auch die Bereitschaft des Besuchers zur kommerziellen Nutzung abgefragt [HLW09]. Die Ergebnisse zeigen zum Einen, dass auch diese Anwendung durch die Rekonstruktion einen deutlichen Mehrwert für den Besucher bietet. Überdies ist dieser mehrheitlich bereit, zusätzlich Geld für die Erfahrung auszugeben.

4 Implementierung

Dieses Kapitel behandelt die Implementierung der Rekonstruktion. Zu Testzwecken wird ein Mockup zum Testen der Technologien erstellt. Außerdem kann dieses durch seine kleineren Ausmaße überall benutzt werden. Im Anschluss wird dann der Prozess zum Erstellen der Vervollständigung der Ruine aufgezeigt und das dazugehörige Level beschrieben.

4.1 Mockup

Zum Testen der benötigten Methoden und Funktionen wird ein Mockup erstellt. Das Ziel dieses Levels ist die Präsentation und das Testen der Features für die Rekonstruktion. Dafür wird eine virtuelle Szene vergleichbar mit einer Rekonstruktion implementiert, welche keine realen Umgebungsfeatures benötigt. Damit kann dieser Level überall gestartet werden, wo genug Platz vorhanden ist. Das Mockup enthält unter anderem ein System zum Sichtbarmachen verschiedener Rekonstruktionsebenen. Außerdem wird die korrekte Ausrichtung der Szene anhand von Raumankern unterstützt. Als Testrekonstruktion wird ein kleines Gebäude ohne Dach mit vier Außenmauern, einer Tür und einigen Stützbalken verwendet.

4.1.1 Sichtebenen

Die Anwendung bietet drei Sichtebenen, die unabhängig voneinander vom Benutzer aktiviert und deaktiviert werden können. Die Aktivierung der Ebenen erfolgt über ein Handmenü (siehe Abbildung 4.1) mit Checkboxen für die drei Optionen. Um auf das Menü zuzugreifen, muss der Nutzer seine flache Hand in den Sichtbereich der Kamera der HoloLens halten, dann erscheint neben der Hand das Menü. Die Interaktion mit demselben erfolgt dann mit der anderen Hand. Diese Methode der Interaktion ermöglicht eine einfache Bedienung des Programms, ohne durch eine feste Platzierung des Menüs im HUD dauerhaft Teile des Displays zu blockieren. Gerade bei großen Gebäuden sorgt das für eine bessere Wahrnehmung der Umgebung.
Die erste und wichtigste Ebene der Anwendung zeigt die Visualisierung. Diese enthält die eigentliche Rekonstruktion der realen Umgebung. Hier werden alle ursprünglich sichtbaren Objekte als virtuelle Nachbildungen ergänzt. Dazu zählen strukturelle Elemente wie Wände und Türen, aber auch Dekorationen, Maschinen und weitere Gegenstände. Im Mockup werden hier, wie in Abbildung 4.2 zu sehen ist, Wände und Türen angezeigt.
Die zweite Ebene stellt die Informationsebene dar. Das Programm enthält zur Informationswiedergabe fest an den Wänden installierte Schautafeln (siehe Abbildung 4.4). Die Schautafeln können verwendet werden, um sowohl textbasierte als auch bildbasierte Informationen zu bestimmten Teilen der Rekonstruktion oder Ruine zu bieten. Dabei ist zu beachten, dass die Informationsebene nur die Informationen zu einer anderen Ebene anzeigt, wenn diese auch sichtbar ist. So werden beispielsweise Informationstafeln, die an einer Wand der virtuellen Rekonstruktion angebracht sind,

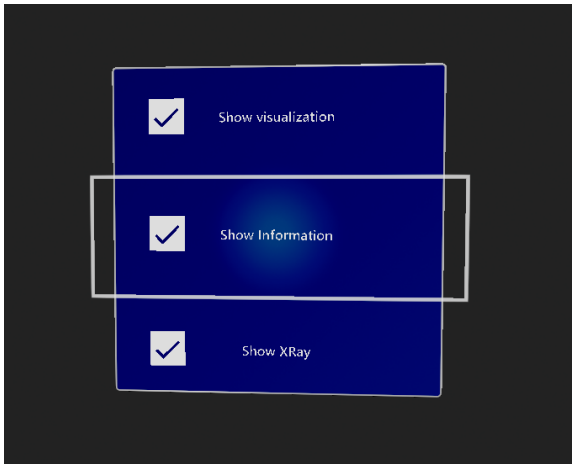


Abbildung 4.1: Ansicht des Menüs zur Änderung der verschiedenen Sichtebenen der Visualisierung.

nur angezeigt, wenn die Visualisierungsebene ebenfalls sichtbar ist.
Die dritte und letzte Ebene der Anwendung wird X-Ray genannt. Diese Ebene stellt normalerweise nicht sichtbare Elemente dar. Das können strukturelle Elemente sein, wie beispielsweise der innere Aufbau einer Wand. Vergrabene Fundamente und Ähnliches werden ebenfalls in dieser Ebene angezeigt . Im Mockup werden hier innere Stützbalken der Wände dargestellt (siehe Abbildung 4.3).

4.1.2 Lokalisierung

Zur akkuraten und fehlerresistenten Verbindung zwischen der realen Ruine und der virtuellen Rekonstruktion gibt es mehrere Varianten. Grundsätzlich existieren für das Projekt zwei verschiedene Welten mit jeweils eigenen kartesischen Koordinatensystemen, die Reale und die Virtuelle.
Das Koordinatensystem der realen Umgebung wird durch die HoloLens definiert, ist rechtshändig und verwendet Meter als Einheit ([the23]). Dieses wird im Anschluss Spatial Koordinatensystem genannt. Dabei befindet sich der Ursprung an der Stelle, an der der Benutzer bei erstmaligem Start der Anwendung steht. Im Anschluss scannt die HoloLens markante Punkte der Umgebung und, wenn der Nutzer sich bewegt oder umsieht, aktualisiert daran die Position und Ausrichtung desselben im Koordinatensystem. Bewegt sich der Nutzer beispielsweise einen Meter vom Ausgangspunkt in der Realität nach vorne, ohne seine Richtung zu verändern, so ist seine Position im Koordinatensystem bei $(0, 0, -1)$.
Das zweite Koordinatensystem stellt die Grundlage für die virtuelle Rekonstruktion dar, in unserem Fall ist dies das integrierte Koordinatensystem der Unreal Engine. Dieses verwendet einen linkshändischen Z-Up Ansatz und Zentimeter als Einheit [Nic]. Der zitierte Artikel beschreibt zwar das Koordinatensystem der Unreal Engine 4, die genannten Punkte gelten aber ebenso für UE5. Innerhalb dieses Koordinatensystems werden dann alle Elemente der virtuellen Rekonstruktion platziert. Ziel der Lokalisierung ist es, eine Transformation zu definieren, die diese beiden Koordinatensysteme verbindet. Nach dem Anwenden der Transformation müssen die virtuellen Objekte an der korrekten Stelle über die Realität projiziert werden. Einen Teil der Transformation stellt dabei die Umrechnung der Einheiten sowie die Rotationen zur Anpassung der verschiedenen Koordinatenachsen dar. Diese ist konstant und im Code des OpenXR Plugins (siehe Abschnitt 2.2.1) bereits definiert.

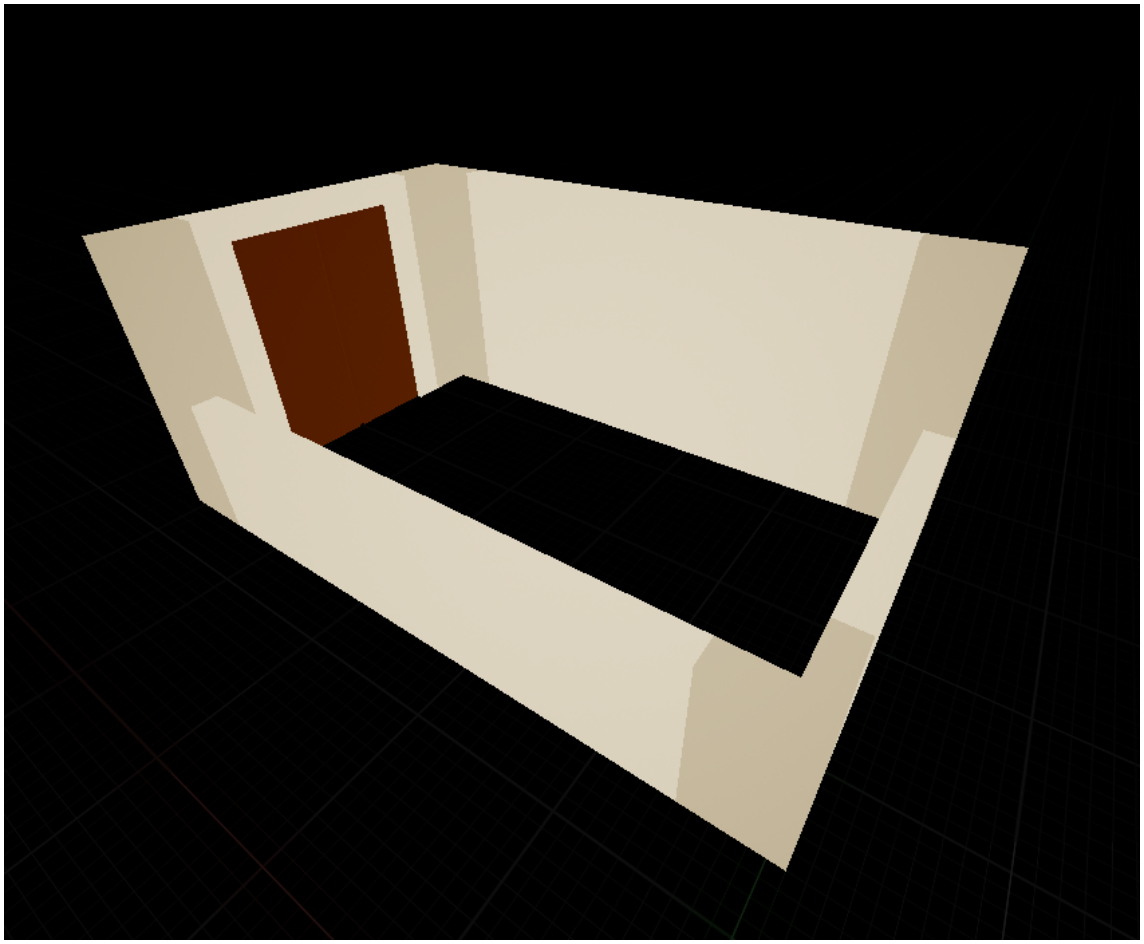


Abbildung 4.2: Visualisierungsebene des Mockups, bestehend aus einem 5m x 3m großen Wandrechteck und einer Tür.

Eine Option zur Erstellung des dynamischen Teils der Transformation funktioniert über den Vergleich zweier Höhenkarten. Die generelle Idee ist es, eine dynamische Höhenkarte der aktuellen Umgebung (siehe Abbildung 4.5a) in einer statischen der realen Umgebung (siehe Abbildung 4.5b) zu finden. Die Transformation zwischen diesen beiden Karten kann dann direkt auf die eigentliche Welttransformation angewendet werden.
Bei der Arbeit mit Texturen bietet es sich in vielen Fällen an, zur Steigerung der Performance Compute-Shader einzusetzen. Diese führen ihre Berechnung parallel auf der Grafikkarte durch. Dabei unterliegen sie gewissen Restriktionen. Zum einen müssen die einzelnen Berechnungen unabhängig voneinander sein, zum anderen ist der Transfer der Daten von der CPU zur GPU und zurück im Vergleich nicht performant.
Für die Umsetzung wird das von der HoloLens automatisch generierte Spatial Mesh als Grundlage herangezogen. Dieses enthält neben den Datenpunkten und Verbindungen zwischen ihnen auch Normalenwerte für jeden Punkt (siehe Abbildung 4.6). Für die Umwandlung dieser Geometrie in eine Höhenkarte müssen die vorhandenen Datenpunkte zuerst jeweils einem korrespondierendem Pixel auf der Textur zugeordnet werden. Je nach Punktdichte im Mesh und Auflösung der Textur kann es vorkommen, dass unterschiedlich viele Datenpunkte auf jeden Pixel abgebildet werden. Zur Berechnung der Verteilung wird ein ComputeShader auf allen Datenpunkten des Meshes ausgeführt.

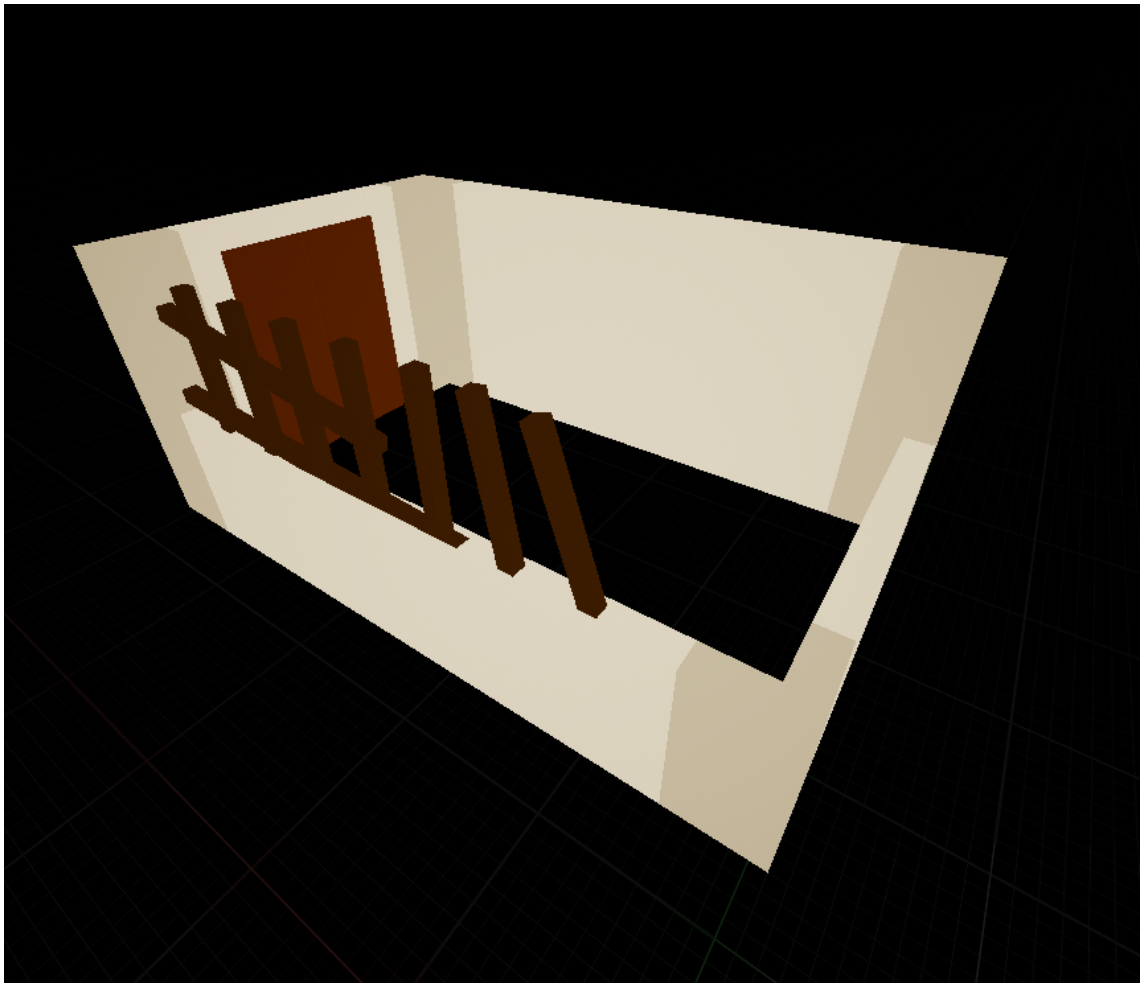


Abbildung 4.3: Die X-Ray Sichtebene wird für Elemente verwendet, welche für die Rekonstruktion normalerweise nicht sichtbar oder relevant sind. Sie bietet weitergehende Informationen über Bautechniken, vergrabene oder verborgene Elemente. Das Mockup zeigt in dieser Ebene den inneren Aufbau einer Wand.

Als Ergebnis gibt es für jeden Pixel auf der Textur eine Liste aller Datenpunkte, die diesen Pixel beeinflussen können (siehe Abbildung 4.9). Zusätzlich werden die minimalen und maximalen X, Y und Z-Werte aller Punkte im Datensatz berechnet. Die X und Y Werte werden benötigt, um die Textur zu skalieren, während die Z Werte genutzt werden, um alle Punkte auf der Z Achse in einen Bereich zwischen 0 und 1 zu bringen.
Im zweiten Schritt wird dann ein weiterer ComputeShader auf jeden Pixel der Textur ausgeführt. Hierfür muss zwischen 3 Fällen unterschieden werden. Ist die Liste der beeinflussenden Datenpunkte für einen Pixel leer, so kann am Ende entweder der Grauwert durch eine Interpolation der Nachbarpixel ermittelt oder einfach auf 0 gesetzt werden. Existiert genau ein Eintrag in der Liste, so wird der Z-Wert dieses Punktes anhand seiner Position im Bereich der minimalen und maximalen Z-Werte des Datensatzes in einen Bereich zwischen 0 und 1 gebracht. Dieser Wert wird dann als Grauwert in der Textur gesetzt. Existieren mehrere beeinflussende Punkte, wird der genommen, welcher am ehesten dem senkrechten Vektor nach oben entspricht. Mathematisch gesehen der, bei

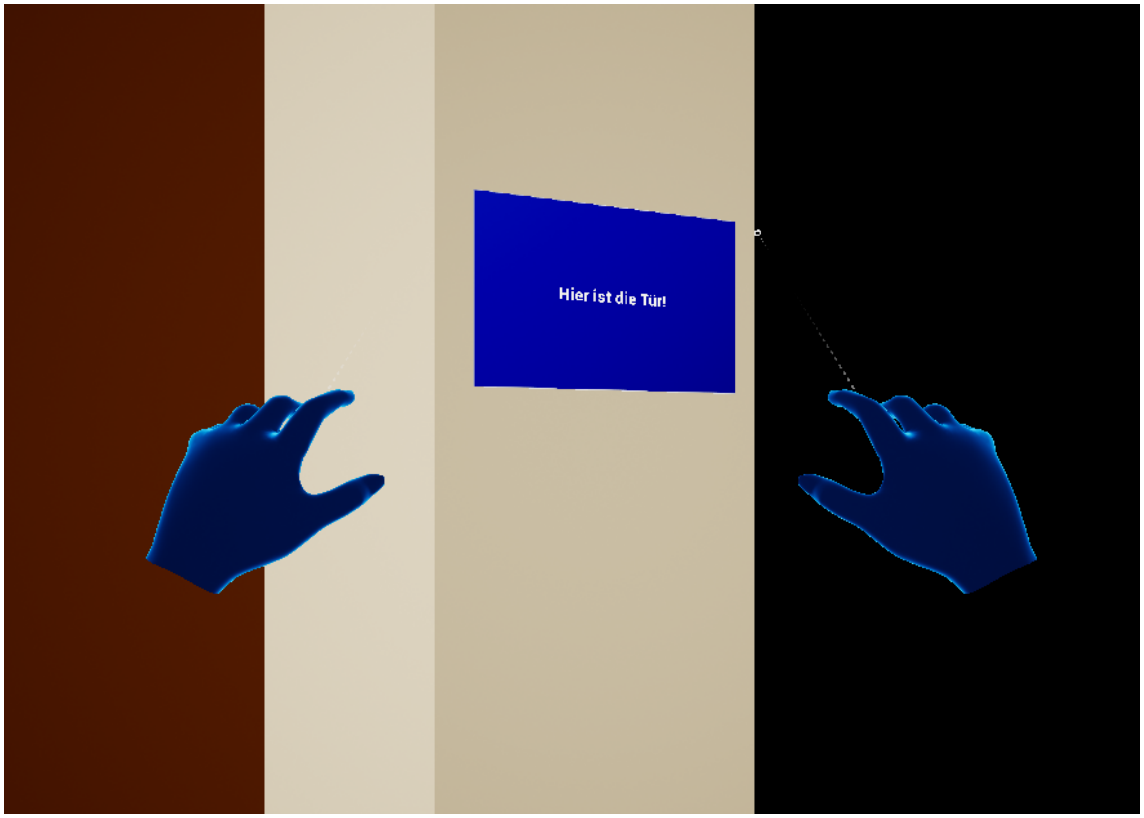
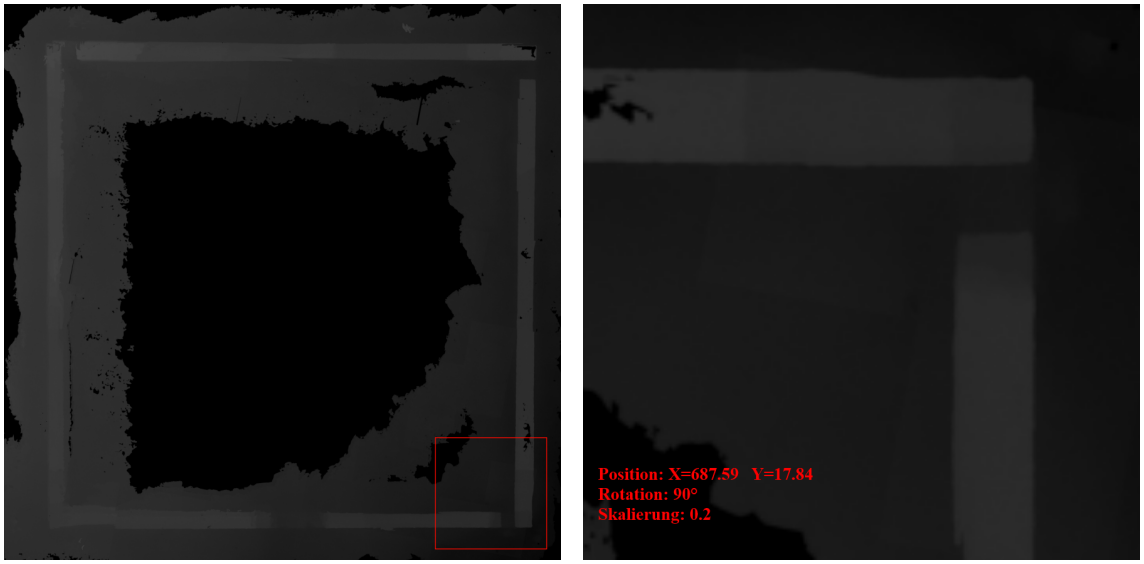


Abbildung 4.4: Zeigt die Platzierung einer Informationstafel an der Außenwand des Mockups.

dem das Skalarprodukt der normalisierten Normale $\vec{n}$ und dem Vektor $\vec{z} = (0, 0, 1)$ am nächsten an 1 herankommt ($max(\vec{n} \cdot \vec{z} =\mid \vec{n} \mid * \mid \vec{z} \mid)$). Es sind auch andere Varianten der Auswahl möglich. Beispielsweise kann auch der Vektor gewählt werden, der in einem gewissen Toleranzbereich t dem senkrechten Vektor gleicht und auf der Z-Achse der höchste ist. Im Anschluss an die Auswahl wird der Grauwert des Pixels anhand des ausgewählten Vektors so bestimmt wie im vorherigen Teil beschrieben.
Das Ergebnis dieser Berechnungen ist eine Höhenkarte. Diese kann im Anschluss noch geglättet werden, um einzelne Ausreißer zu eliminieren. Ausreißer können entstehen, wenn für einen Pixel nur ein Punkt existiert, welcher aber eigentlich schlecht geeignet ist. Beispielsweise ein Punkt mit nach unten gerichteter Normale, welcher einen Teil einer Decke repräsentiert.
Die erstellte Höhenkarte wird im nächsten Schritt mit einer fest definierten Höhenkarte des Ruinengebiets abgeglichen. Durch eine Bilderkennung wird die erstellte Höhenkarte in der statischen erkannt und eine Transformation zwischen diesen beiden Karten errechnet (siehe Abbildung 4.5b). Diese Berechnung kann nur eingeschränkt parallel durchgeführt werden und ist deswegen sehr intensiv seitens der Performance.
Der Vorteil dieser Technik ist die Genauigkeit bei Bewegungen des Nutzers. Diese hängt jedoch stark von der Auflösung der Textur und dem Detailgrad des Spatial Meshes ab. Durch andauerndes Aktualisieren wird so aber eine genaue Platzierung der virtuellen Komponenten erreicht. Aufgrund der vergleichsweise geringen Hardwareleistung der HoloLens ist es allerdings nicht möglich, dauerhaft diese Berechnungen durchzuführen, weshalb dieses Konzept für die Realisierung der Anwendungen verworfen wird.



(a) Höhenkarte des römischen Hauses. Das rote Quadrat zeigt den Ausschnitt, der in Abbildung 4.5b transformiert dargestellt ist.

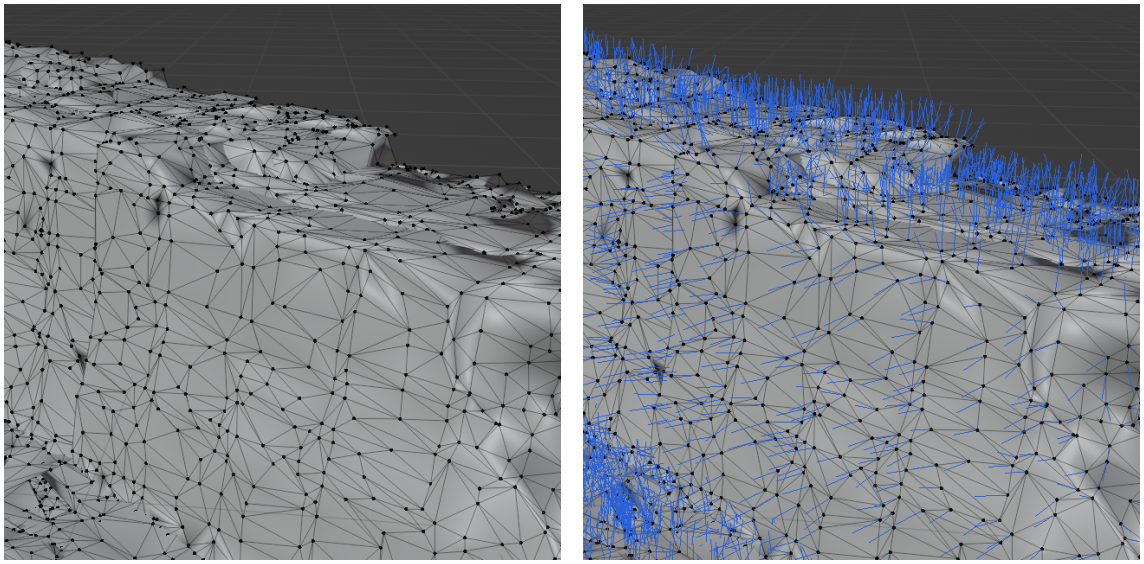
(b) Darstellung des transformierten Ausschnitts der Höhenkarte aus Abbildung 4.5a. Die Werte der Transformation sind unten links dargestellt.

Abbildung 4.5: Darstellung des Codes zur Berechnung der Transformation, beschrieben im ersten Teil von Abschnitt 4.1.2. Abbildung 4.5a zeigt die statische Höhenkarte der realen Umgebung, Abbildung 4.5b die aktuelle dynamisch erstellte Karte.

Wie auch für die spätere Implementierung der realen Ruine werden für das Mockup Raumanker basierend auf QR-Codes verwendet. In Abbildung 4.7 sind die vier QR-Codes abgebildet, welche für dieses Projekt verwendet werden. Abbildung 4.8 verdeutlicht die Korrelation der QR-Codes zu ihrer Position in der virtuellen Szene. Wird ein QR-Code in der realen Welt durch die Kamera der HoloLens erkannt, erfolgt eine Prüfung ob ein Weltanker mit dem Wert dieses QR-Codes im Level vorhanden ist. Stimmt er mit einem der Weltanker überein, wird dessen reale Position im Spatial Koordinatensystem auf die Position des QR-Codes gesetzt. Daraus berechnet der Anker eine Transformationsmatrix zwischen seiner Position in der virtuellen Szene und der zugewiesenen Position im Spatial Koordinatensystem. Durch die Positionierung mehrerer Anker berechnet des Programm eine Interpolation zwischen den verschiedenen Weltankern anhand der aktuellen Nutzerposition. Befindet sich der Nutzer beispielsweise in der Mitte zwischen zwei Weltankern, wird die Position des Hologramms durch eine Interpolation dieser beiden Positionen bestimmt. Steht der Nutzer genau auf einem Anker, wird dessen Position als Transformationsgrundlage genommen. Im Falle des Mockups werden die vier Anker an den vier Ecken der Szene platziert.

4.1.3 Verwendung

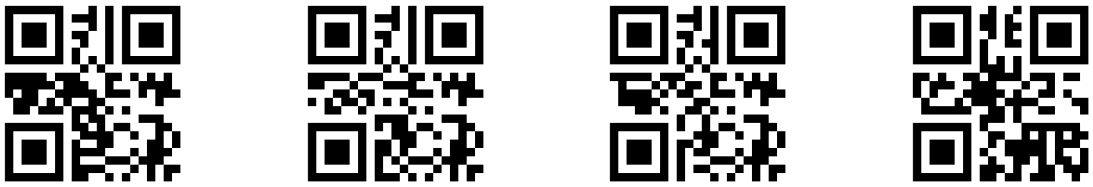
Um das Mockup zu verwenden, muss die Anwendung gestartet und dann im ersten Menü der Reiter „Mockup“ ausgewählt werden (siehe Abbildung 4.10). Wenn dieses Level zum ersten Mal in einer realen Umgebung geöffnet wird, wird die Geometrie des Levels auf Augenhöhe des Nutzers angezeigt. Dadurch scheint das Gebäude zu schweben. Die Kopfposition des Benutzers ist relevant für die ursprüngliche Ausrichtung des Levels (siehe Abbildung 4.11). Wenn das Level in der



(a) Von der HoloLens 2 generiertes Spatial Mesh einer Mauer der Ruine. Zu sehen sind die einzelnen Datenpunkte sowie die Kanten und Flächen.

(b) Gleiches Mesh wie in Abbildung 4.6a. Zusätzlich sind noch die Normalen der einzelnen Datenpunkte als blaue Linien zu sehen.

Abbildung 4.6: Beispielhafte Darstellung der enthaltenen Daten im von der HoloLens 2 generierten Spatial Mesh.



(a) BA_LG_2023_00 **(b)** BA_LG_2023_10 **(c)** BA_LG_2023_01 **(d)** BA_LG_2023_11

Abbildung 4.7: Darstellung der vier für das Projekt verwendeten QR-Codes. Die Koordinaten 00 bis 11 stehen hierbei für die X- und Y-Koordinate, an welcher dieser QR-Code platizert werden muss.

aktuellen Umgebung zuvor bereits gestartet wurde, wird sich die HoloLens nach einigen Sekunden orientieren und das Hologramm auf die letzte bekannte Position setzen.
Zum Start des Levels sind immer alle Sichtebenen (Abschnitt 4.1.1) aktiviert.
Zum Platzieren des Hologramms werden die QR-Codes benötigt. Für das beste Ergebnis sollten sie so ausgelegt werden, dass die Abstände zwischen den Codes den Abständen der Weltanker in der virtuellen Szene entsprechen. Die Anker in der Szene bilden ein Rechteck mit einer Kantenlänge in X-Richtung von 5 Metern und in Y-Richtung von 3 Metern (Abbildung 4.8). Je genauer diese Maße auch in der realen Welt eingehalten werden, desto weniger Verschiebungen des Hologramms durch Interpolation treten auf, wenn man sich von einem Anker zum Nächsten bewegt. Ist das Mockup einmal platziert, können die QR-Codes entfernt werden. Um die Sichtbarkeit der verschiedenen Visualisierungsebenen zu ändern, muss der Nutzer die flache Hand hochkant vor sich halten. Dadurch erscheint ein Menü, in welchem die Checkboxen für die entsprechende Ebene aktiviert

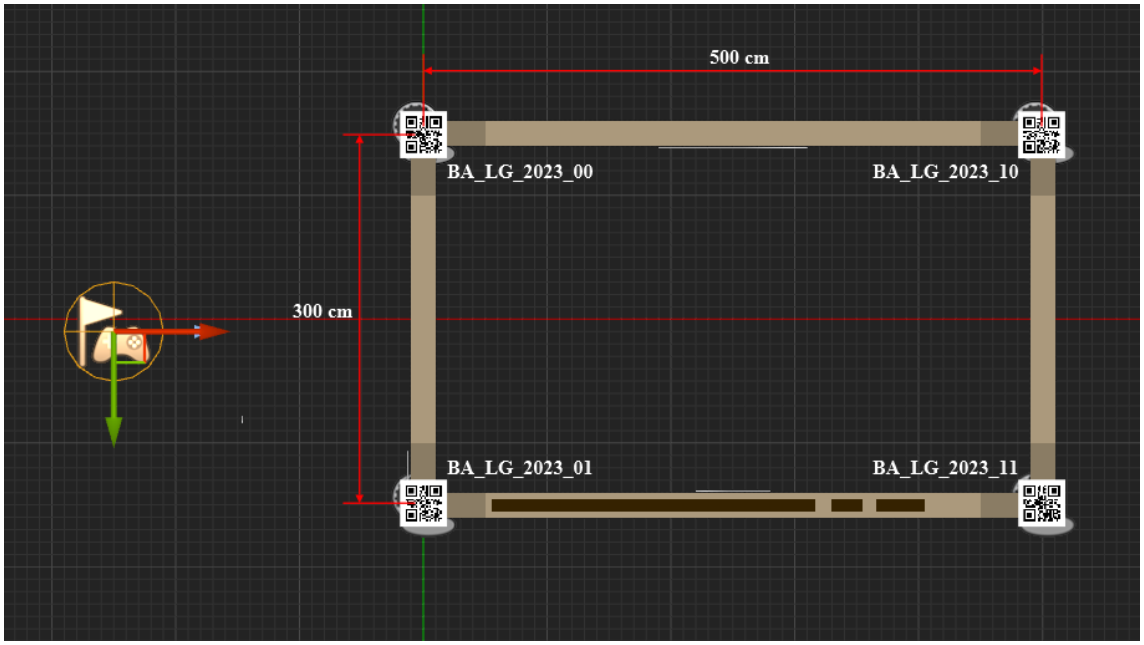
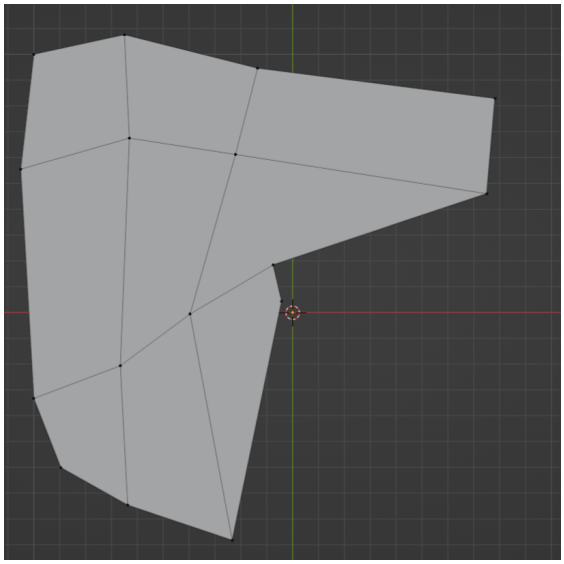


Abbildung 4.8: Visualisiert die Relation der QR-Code Platzierung mit der virtuellen Szene. Die Szene zeigt das Mockup wie in Abbildung 4.3, orientiert, sodass der rote Pfeil von außen auf die Tür zeigt. Die Maße zeigen die Abstände zwischen den QR Codes in cm. Diese Abstände sollten beim Auslegen der Codes in der echten Welt möglichst genau eingehalten werden.

beziehungsweise deaktiviert werden können. Um zum Ende wieder aus dem Level in das Hauptmenü zu wechseln, muss der Nutzer nach unten schauen und dort den Menü-Button (Abbildung 4.12) antippen.

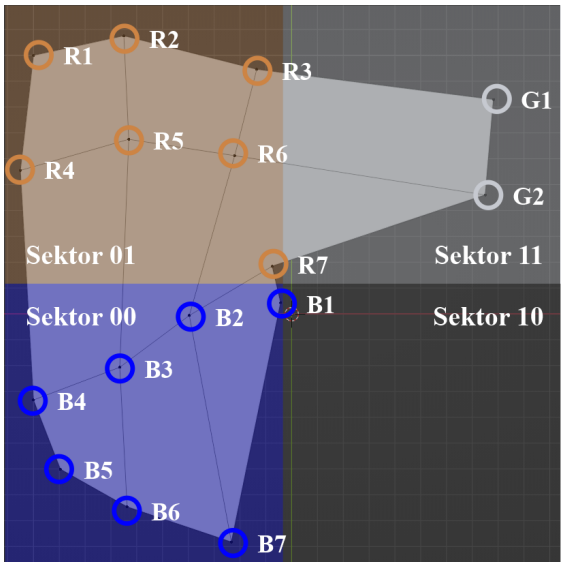
4.2 Reale Ruine

Als Beispielruine für diese Arbeit wurde das römische Haus in Stuttgart gewählt (siehe Abschnitt 2.3). Einerseits aufgrund ihrer Nähe, sodass Tests und Anpassungen einfach und kostengünstig durchgeführt werden können. Zum anderen aufgrund der Einfachheit der Geometrie der Grundstruktur. Des Weiteren ist die vermutete ursprüngliche Konstruktion unkompliziert (siehe Abbildung 4.13). Sie bietet darüber hinaus einige Besonderheiten. So zum Beispiel den innenliegenden Porticus, welcher die Reste der Außenmauern mit im Boden vergrabenen, aber durch die X-Ray Ebene sichtbar gemachten, Fundamenten verbindet. Da die integrierten Recheneinheiten der HoloLens nur begrenzte Leistung liefern, ist es außerdem wichtig, die Polygonanzahl der Szene gering zu halten. Die erstellte Rekonstruktion erlaubt das, ohne wichtige Details entfernen zu müssen.



(a) Beispielgeometrie mit mehreren Datenpunkten.

(b) Beispieltextur mit 4 Pixeln. Zur besseren Übersicht sind die vier Pixel der Textur farblich gekennzeichnet.



(c) Zuordnung der einzelnen Datenpunkte zu einem Pixelbereich der Textur.

(d) Listen der beeinflussenden Datenpunkte für jeden Pixel der Textur.

Abbildung 4.9: Darstellung des Grundprinzips des Computeshaders, der die beeinflussenden Datenpunkte für jeden Pixel in der Textur bestimmt. Die in Abbildung 4.9d abgebildeten Listen sind das Ergebnis dieses Shaders.

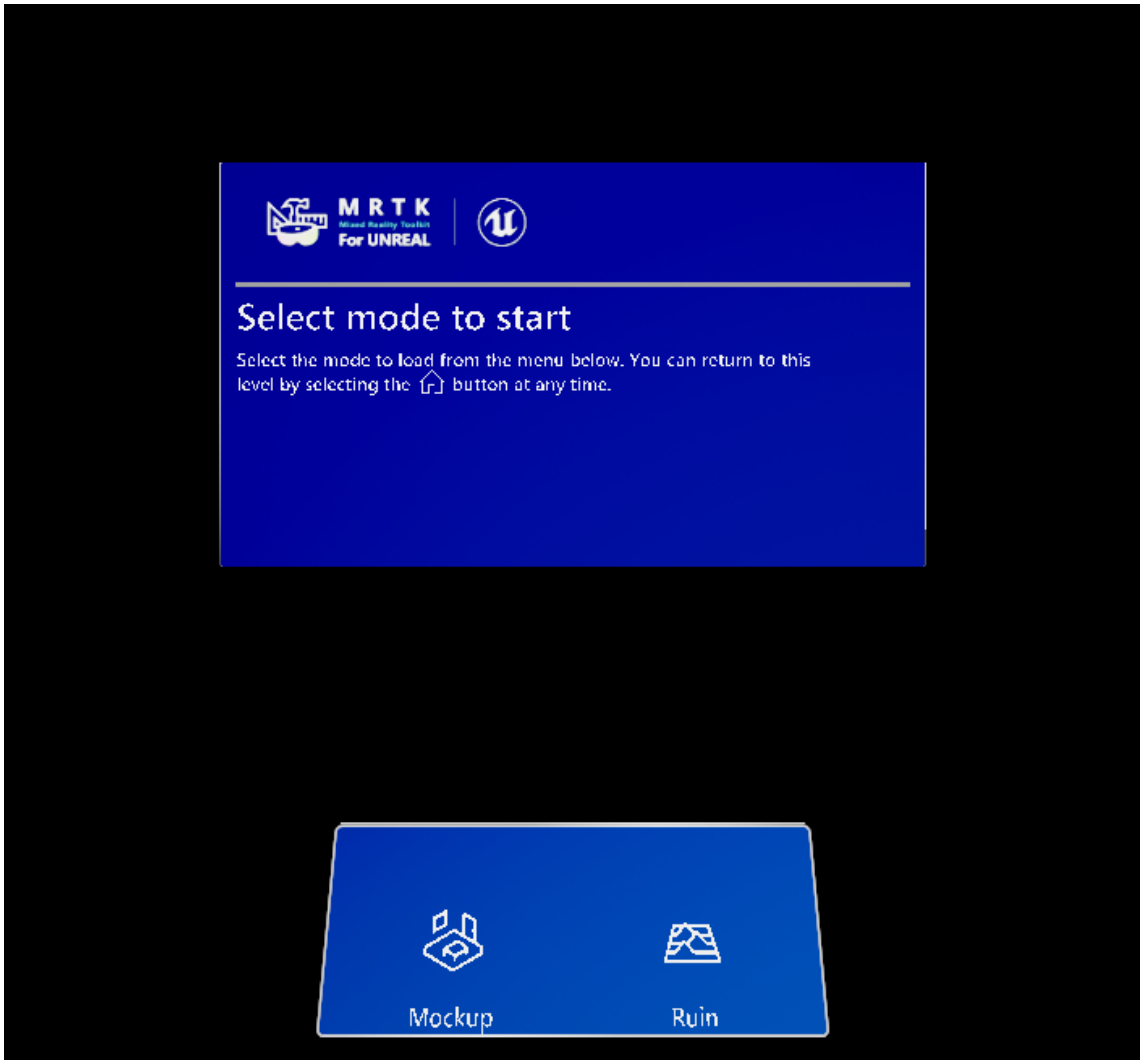


Abbildung 4.10: Zeigt das Startmenü des Projektes. Um zur Visualisierung des Mockups zu gelangen muss der Mockup Button unten links gedrückt werden, für die Ruine der Button unten rechts. Der Nutzer kann jederzeit zu diesem Menü zurückkehren um einen anderen Level auszuwählen.

4.2.1 Scan

Der erste Schritt zur Erstellung einer virtuellen Rekonstruktion ist es, eine möglichst genaue virtuelle Abbildung der existierenden Ruine zu bekommen. Wie in Kapitel 3 bereits erwähnt wurde, wählen viele ähnliche Projekte den Ansatz, die Ruine komplett virtuell zu rekonstruieren. Dabei werden auch die noch real existierenden Teile überdeckt. Um solche vollständigen Rekonstruktionen zu erstellen sind die Abmessungen der realen Ruine wichtig, allerdings nicht Messdaten über genaue Mauerhöhen und Einbuchtungen. Da in unserem Projekt der virtuelle Anteil auf die existieren Ruinen aufgesetzt werden soll, ist eine genauere Vermessung der gesamten Maueroberfläche erforderlich. Zu diesem Zweck werden zwei Ansätze getestet.
Die erste Möglichkeit ist die Verwendung von dem von der HoloLens standardmäßig generierte

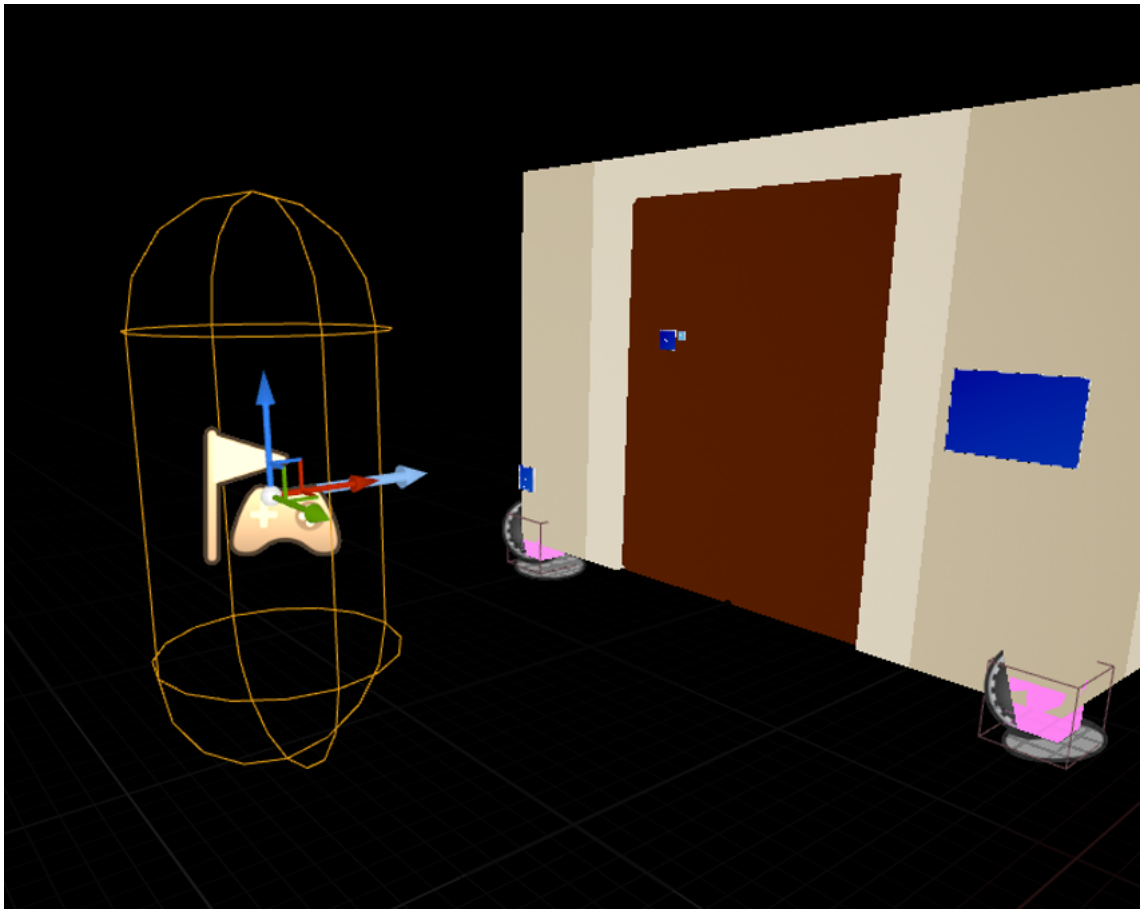


Abbildung 4.11: Darstellung des Levels mit dem Spawnpoint links im Bild. Die Kopfhöhe im Level ist die Mitte des Gizmos.

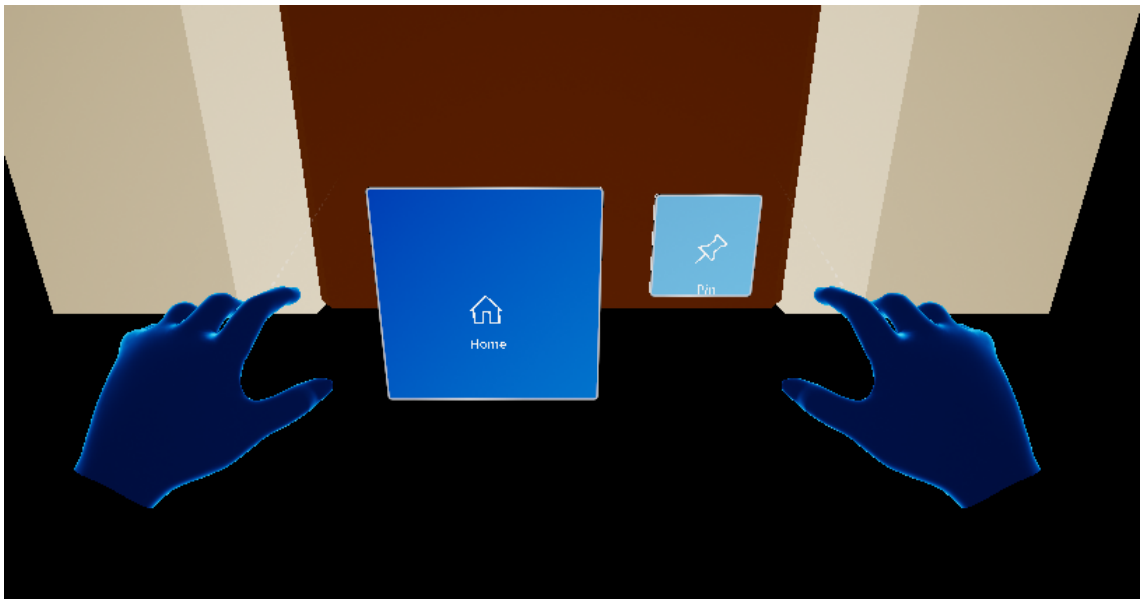


Abbildung 4.12: Zeigt das Menü, mit welchem der Benutzer zurück zur Levelauswahl gelangt.

Abbildung 4.13: Schautafel am Ort des römischen Hauses in Stuttgart. Zu sehen ist eine Übersicht über vorhandene Ruinenelemente sowie eine mögliche Rekonstruktion. Diese dient als Grundlage der virtuellen Rekonstruktion im Rahmen dieser Arbeit (Quelle: TimoMM, https://www.evolution-mensch.de/Anthropologie/Datei:Schautafel_Roemisches_Haus_Rotwildpark.jpg, Lizenz: Creative-Commons-Lizenz „Namensnennung 3.0 nicht portiert“, https://creativecommons.org/licenses/by/3.0/deed.de))

Spatial Mesh. Dieser Ansatz funktioniert sehr gut in geschlossenen Räumen mit begrenzten Maßen, vielen Orientierungspunkten und gleichmäßiger, nicht zu heller Beleuchtung. Leider sind diese Punkte im Wald bei Sonnenschein nur begrenzt gegeben. Das Ergebnis des Scans hat deswegen einige Probleme und Fehler, von denen die markantesten im Anschluss beschrieben werden.
In den Teilen der Ruine, die von der Sonne bestrahlt werden, funktioniert die Generierung der Geometrie oftmals entweder gar nicht oder nur sehr langsam und mit viel Bewegung des Nutzers. Genaue Gründe konnten nicht sicher benannt werden, es ist allerdings wahrscheinlich, dass die Infrarotstrahlung der Sonne mit denen des Scanners der HoloLens interferiert. Das Mesh dieser Bereiche weist deswegen oft Löcher oder in der Realität nicht vorhandene Höhenunterschiede auf. Weiterhin stellt die Größe der Ruine ein Problem dar. Ein vollständiges Scannen der vorhandenen Mauern von einem statischen Punkt aus ist nicht möglich. Stattdessen ist es notwendig, die Außenmauern während des Scans abzulaufen. Aufgrund von Ungenauigkeiten in der Mesherstellung und dem Tracking führt das allerdings zu Abweichungen. In den durchgeführten Scans entstehen immer Höhenunterschiede und seitliche Verschiebungen, wenn man nach einem Rundgang wieder an bereits

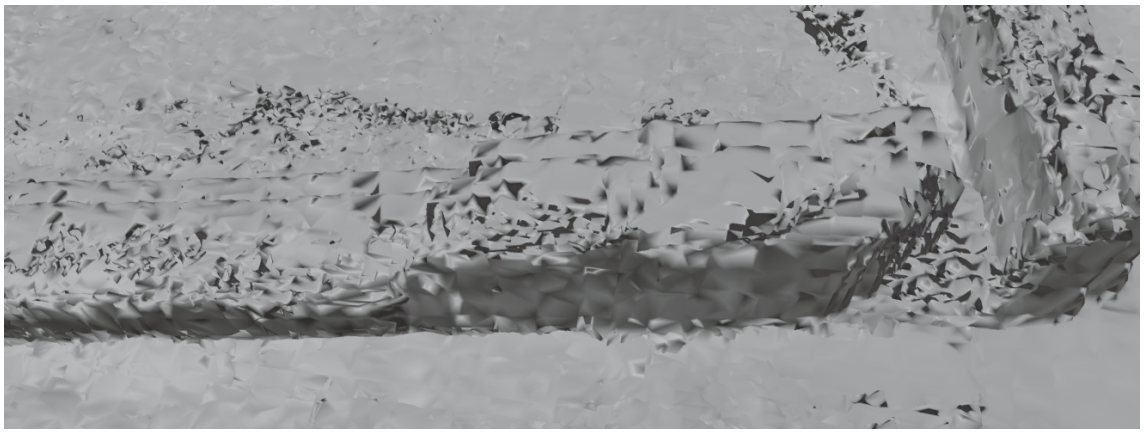
gescannte Mauerteile kommt. Abbildung 4.14 zeigt das am Beispiel der nordwestlichen Ecke. Diese Abweichungen betragen im speziellen Fall bis zu 50 cm. Als Gründe für dieses Ergebnis kommen viele Ursachen in Frage, ein großes Problem sind vermutlich fehlende markante Bereiche der Ruine. Im Grunde besteht das Gebiet nur aus ähnlich aussehenden Mauern mit gelegentlichen Türöffnungen (siehe Abbildung 2.1). Das sorgt dafür, dass die Brille zeitweise die aktuelle Umgebung nicht als bereits gescannt erkennt. Vor allem nach einem Neustart oder schnellen Bewegungen geht der Kontext der Umgebung schnell verloren. Der flache Innenraum der Ruine ist vermutlich ein weiterer Grund dafür, dass die Brille den Kontext verliert. Eine Möglichkeit, die Qualität des Scans zu verbessern, ist es, künstliche Marker zu platzieren. Dadurch fällt es dem Algorithmus leichter den Kontext zu behalten.
Zuletzt lässt sich der Scan der Ruine nicht als eine Datei exportieren. Stattdessen muss die Datei in vier Bereiche aufgeteilt werden, welche dann exportiert werden können. Diese vier Dateien können zwar anschließend in Blender Abschnitt 2.2.2 wieder zusammengefügt werden, dabei kann es allerdings potentiell zu Abweichungen kommen.
Das Ergebnis dieses Scans ist nur eingeschränkt verwendbar. Die grundsätzliche Struktur ist zwar vorhanden, die großen Ungenauigkeiten (siehe Abbildung 4.14) erfordern es jedoch, das eigentliche virtuelle Modell nur grob zu erstellen und dann vor Ort zu testen und anzupassen. Da das Anpassen nicht in der Anwendung direkt vorgenommen werden kann, ist diese Variante zeitintensiv und schwer umsetzbar.
Die zweite Variante ist die Anwendung eines Laserscanners. Dieser erlaubt die Erstellung einer äußerst genauen Repräsentation der Ruine, und das nahezu unabhängig von äußeren Lichtverhältnissen oder anderen Störfaktoren. Dafür wird der Scanner in der Mitte der Ruine platziert. Abbildung 4.15 zeigt die Platzierung des Scanners und das erzeugte Ergebnis. Das Ergebnis unterscheidet sich deutlich von dem Spatial Mesh der HoloLens. Der Scanner liefert keine verbundene Geometrie, stattdessen wird eine Punktwolke erstellt. Jeder ausgesendete Laserstrahl erzeugt bei Kontakt mit einem Objekt einen Punkt im Raum. Dadurch lässt sich zwar eine sehr genaue Approximation der realen Gegebenheiten erreichen, es müssen jedoch einige zusätzliche Schritte ausgeführt werden, bevor die Rohdaten verwendet werden können. Weiterhin ist die Dichte der Punktwolke sehr ungleich. Je nach Winkel der realen Objekte zum Scanner nimmt die Dichte zu oder ab. Zusätzlich spielt auch die Entfernung vom Scanner eine Rolle. Je näher sich ein Objekt am Scanner befindet, desto höher ist die Punktdichte, mit der dieses Objekt abgebildet wird. Diese Eigenschaft führt auch zu der erkennbaren strahlenförmigen Ausbreitung der Datenpunkte. Abbildung 4.16 zeigt den Einfluss der Entfernung und des Winkels der zu scannenden Geometrie auf die Punktdichte. Diese Charakteristiken bewirken, dass zum einen die Punktdichte in der Mitte der Ruine auf der Grasfläche am höchsten ist. Für die Rekonstruktion spielt dieser Bereich jedoch fast keine Rolle. Außerdem sind die Innenseiten der Mauern sehr detailgetreu vorhanden, die Oberseite aber nur sehr beschränkt (Abbildung 4.17). Für die anschließende Rekonstruktion ist wie in Abschnitt 4.2.2 beschrieben die Oberseite der Mauern relevant. Trotz dieser Probleme wird die Restaurierung mit den Daten des Laserscanners durchgeführt, vor allem aufgrund seiner hohen Präzision im Vergleich zum Spatial Mesh der HoloLens.
Der Dateityp des Exports lässt sich direkt in Blender importieren. Eine Besonderheit ist, dass der Laserscanner seine Daten in Millimeter abspeichert, Blender aber standardmäßig mit Metern als Standardeinheit arbeitet. Das führt dazu, dass die Punktwolke nach dem ersten Import einen Durchmesser von etwa 26 km hat. Um diesen Fehler zu beheben gibt es in Blender die Möglichkeit, die Standardeinheit auf Millimeter zu ändern. Aufgrund von möglichen späteren Exportproblemen in Verbindung mit dem Standardmaß der Unreal Engine, Zentimeter, ist ein Skalieren aller Punkte mit dem Faktor 0.001 hier die effizientere Wahl.

Die erhaltende Punktwolke hat etwa 64 Millionen Datenpunkte, wodurch es zu Hardwareproblemen kommen kann. In meinem Fall kam es zu Performanceproblemen aufgrund des Arbeitsspeichers. Zum Beheben der Probleme wird im ersten Schritt die Anzahl der Datenpunkte in der Datei verringert. Dies funktioniert beispielsweise mithilfe des Mergetools, welches aus der Menge der ausgewählten Datenpunkte alle zu einem zusammenfügt, die im Umkreis einer gewissen Distanz X liegen. Für die erste Verringerung der Daten im Zentrum des Scans, welcher die Grasfläche darstellt, wird für X ein Wert von 0.2m genommen. Abbildung 4.18 zeigt diesen ersten Schritt und die Auswirkungen auf die Zahl der Datenpunkte. Da die genaue Geometrie der Außenmauern sehr wichtig und die Datendichte standardmäßig geringer ist, wird hier für X ein Wert von 0.05m angenommen. Die Unterschiede zwischen der groben Auflösung im Inneren und der Feineren an den Mauern erkennt man ebenfalls in Abbildung 4.18.

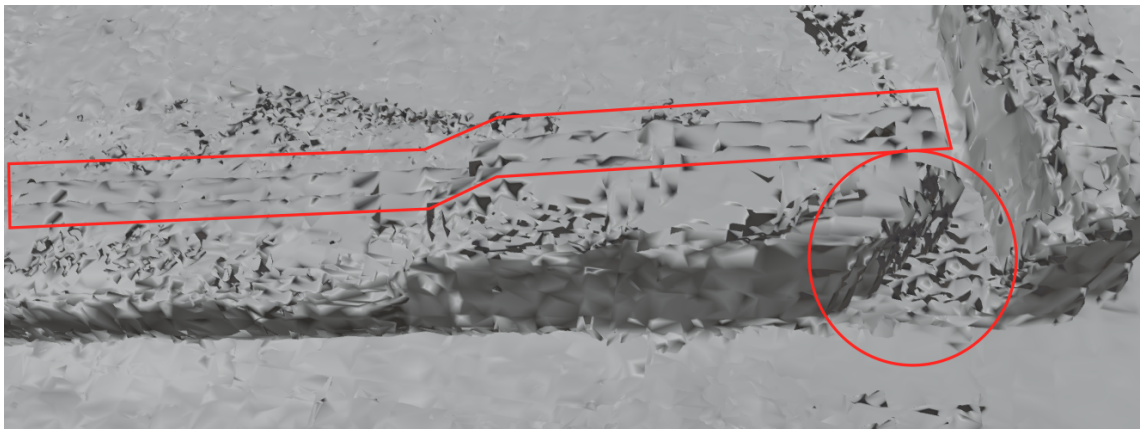
4.2.2 Virtuelle Rekonstruktion

Die Art der Rekonstruktion orientiert sich an der Darstellung des Gebäudes auf der Schautafel, welche vor Ort ausgestellt ist (siehe Abbildung 4.13). Wie im Abschnitt 4.2.1 erwähnt, stellt die Oberseite der Mauern die Grundlage für die Rekonstruktion dar. Der Grund dafür ist die Art der Ruine, bei der nur Mauerteile über den vorhandenen Mauern fehlen. Der nächste Schritt der Rekonstruktion ist die Aufteilung der Punktwolke in drei Bereiche, nämlich den Boden (Abbildung 4.19a), die vertikalen Mauerseiten (Abbildung 4.19b) und die Maueroberseite (Abbildung 4.19c). Jeder dieser Bereiche spielt für einen anderen Teil der Rekonstruktion eine Rolle.
Die Maueroberseite dient gemeinsam mit den Mauerseiten als Grundlage für die virtuelle Ergänzung des Gebäudes. Aufgrund von geringfügigen Schwankungen in der Punktwolke und einer, besonders außen an den Mauern, teilweise sehr begrenzten Punktdichte, können diese Punkte nicht direkt zu einem Mesh zusammengefügt werden. Stattdessen wird über diese Punkte eine Mesh-Ebene gelegt, welche in Länge, Breite und Höhe die Kontur der Mauerkrone repräsentiert (Abbildung 4.20a). Diese Ebene wird dann nach oben zu einem dreidimensionalen Körper erweitert, mit einer gemeinsamen Höhe. Damit ist die Rekonstruktion der Wand abgeschlossen (Abbildung 4.20b).
Aufgrund von heute nicht mehr sichtbaren Fundamenten parallel entlang der Wände im Innenbereich der Ruine wird vermutet, dass sich rund um den Innenbereich ein überdachter Bereich befunden hat [Pla05]. Dieser Porticus genannte Gang bestand vermutlich aus Holzbalken und wurde mit Ziegeln gedeckt. Ähnliche Gebäudestrukturen sind in vielen römischen Bauwerken zu finden [21]. Um diesen mit möglichst wenig Geometrie realitätsnah nachzubilden werden nur zwei Einzelteile in Blender als Modell erstellt. Zum einen der Eckbereich (Abbildung 4.21b) und zum anderen ein Teil des geraden Bereichs (Abbildung 4.21a). Diese Einzelteile werden später mithilfe der Unreal Engine mehrfach im Level eingesetzt (siehe Abschnitt 4.2.3).
Zusätzlich zu diesen großen Teilen werden noch Modelle für die Türen erstellt. Aufgrund der Gegebenheiten an der Ruine werden zwei verschiedene Modelle benötigt. Eine Doppeltür mit zwei Metern Breite, und drei identische Türen mit jeweils einem Meter Breite.
Außerdem gibt es noch Modelle für die vorhandenen Fundamente im Boden. Da es keine genauen Informationen zu deren Aussehen gibt, ist die Form einfach gehalten. Die Funktion wird im späteren Level mit einer Informationstafel beschrieben.
Das letzte benötigte Modell stellt eine im Boden entdeckte, heute aber auch nicht mehr real sichtbare Grube dar. Diese war den Ausgrabungen zufolge aus Stein und mit einer Holzschalung eingefasst. Sie enthielt Scherben verschiedenster Gefäße aus dem frühen 2. Jahrhundert.
Der Teil der Punktwolke, der den Boden repräsentiert, wird im Anschluss für die Platzierung von

(a) Reales Aussehen des betreffenden Mauerabschnitts.



(b) Ergebnis des Scans der HoloLens.



(c) Roter Kasten: Seitliche Verschiebung der Mauern. Roter Kreis: Mehrere verschiedene Abstufungen der Ecke im Scan, im Abbildung 4.14a ist zu sehen dass die Ecke real keine Abstufungen aufweist

Abbildung 4.14: Visualisierung der Verschiebungen, welche im Endergebnis des HoloLensscans zu sehen sind. Gescannt wurde durch ein Ablaufen der Mauern. Nachdem man wieder am Start ankommt hat sich das Mesh durch Ungenauigkeiten verschoben. Dadurch entstehen die verschiedenen Level innerhalb des Scans.

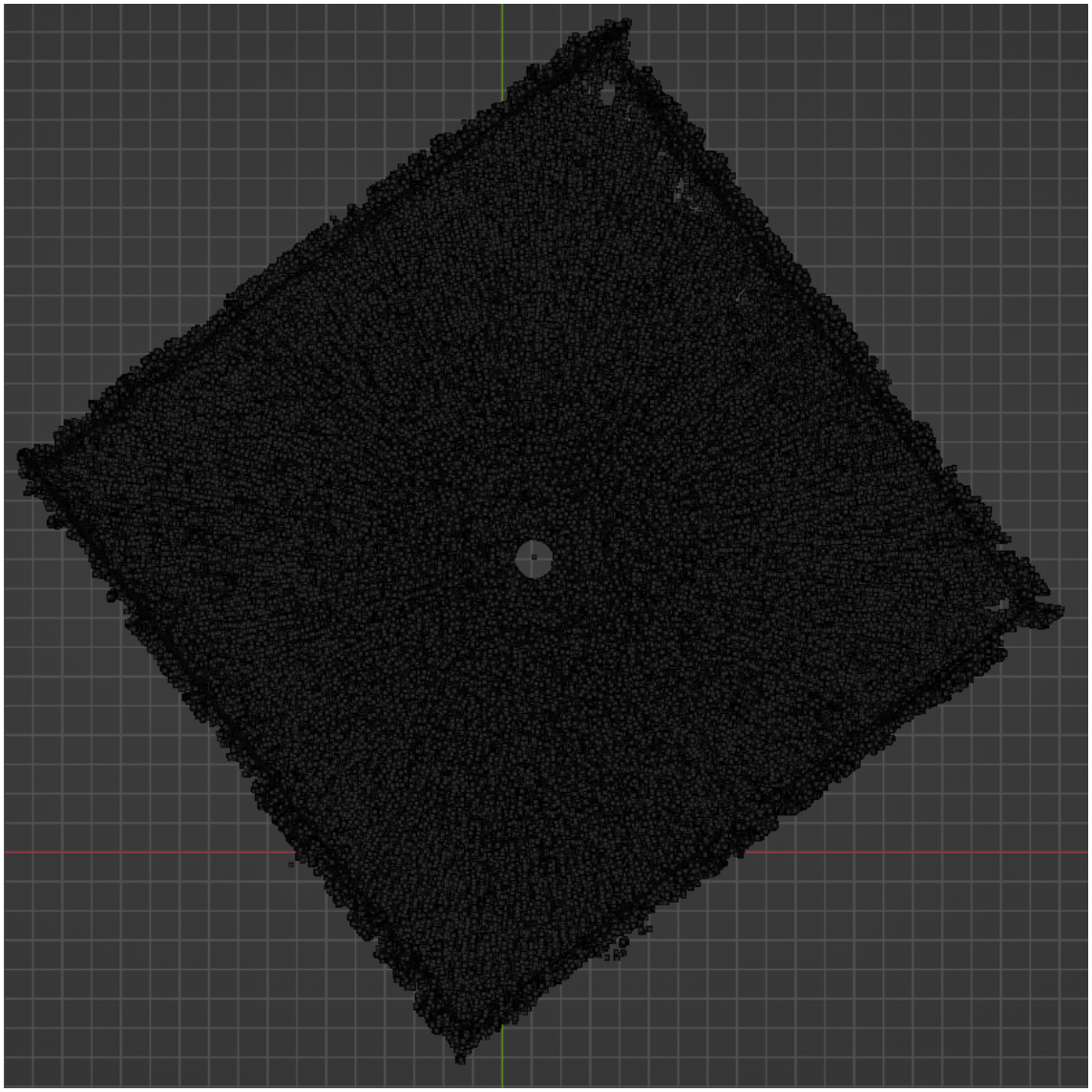
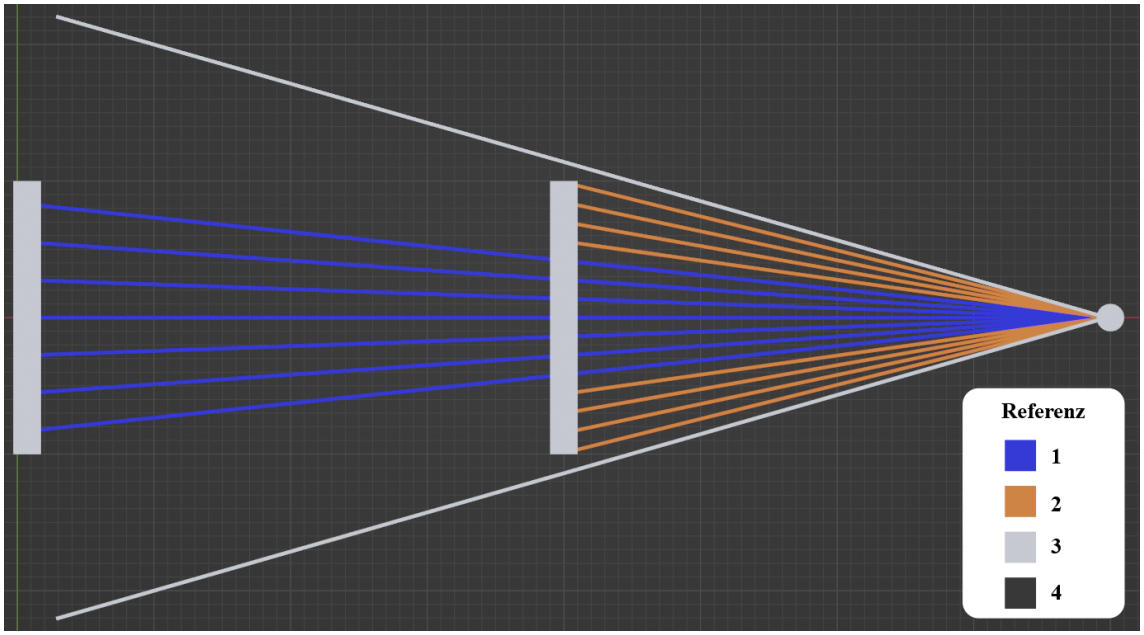
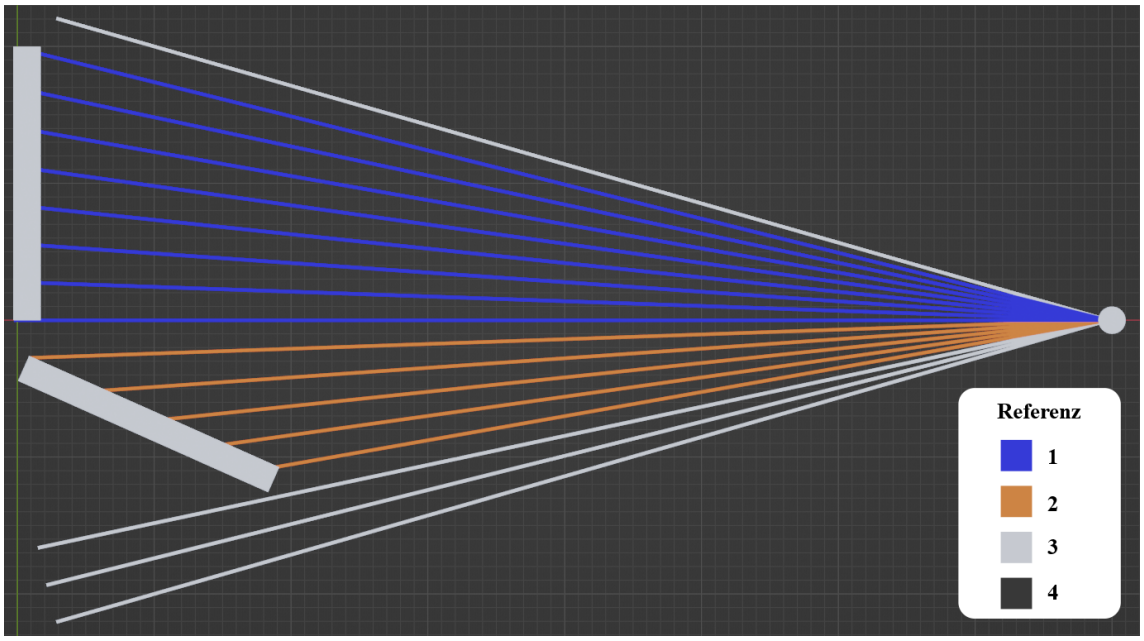


Abbildung 4.15: Die Abbildung zeigt die Rohdaten des Laserscans. Diese bestehen nur aus einzelnen Datenpunkten. Der leere Kreis in der Mitte repräsentiert die Position des Laserscanners, da dieser die Oberfläche unterhalb nicht scannen kann. Weiterhin sind außen die Innenseiten der Außenmauern zu erkennen. Zusätzlich lässt sich auch die Ausrichtung der Ruine erkennen, da der Scanner die Datenpunkte so abspeichert, dass Norden oben liegt. Diese Ausrichtung passt auch zum Schaubild Abbildung 4.13.



(a) Auswirkung der Distanz eines Objektes zum Scanner auf die Punktdichte. Das nähere Objekt wird durch deutlich mehr Strahlen (Blau und Orange, Referenz 1 und 2) abgebildet als das weiter weg (Blaue Strahlen, Referenz 1).



(b) Auswirkung des Winkels eines Objektes relativ zum Scanner auf die Punktdichte. Das gerade Objekt wird durch deutlich mehr Strahlen (Blau, Referenz 1) abgebildet als das Rotierte (Orange Strahlen, Referenz 2).

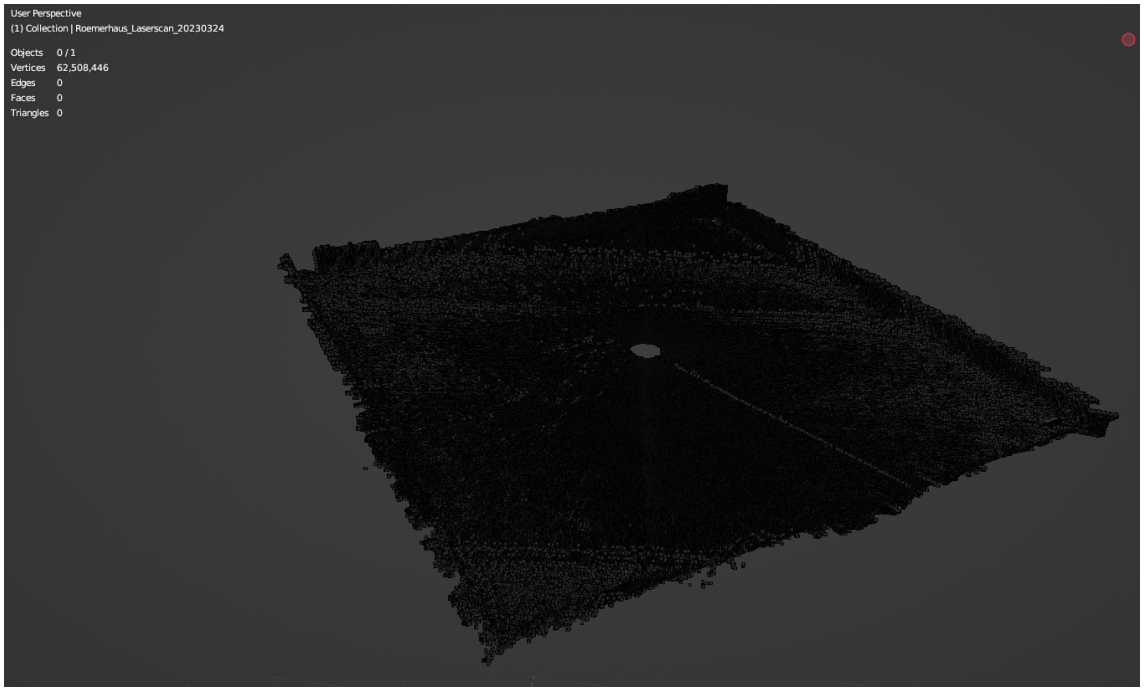
Abbildung 4.16: Diese Grafik visualisiert Auswirkungen verschiedener Transformation von Objekten auf die Punktdichte, mit welcher sie im Ergebnis des Laserscanners abgebildet werden.

Abbildung 4.17: Unterschiedliche Punktdichten an den senkrechten Wänden und den Oberseiten der Mauern.

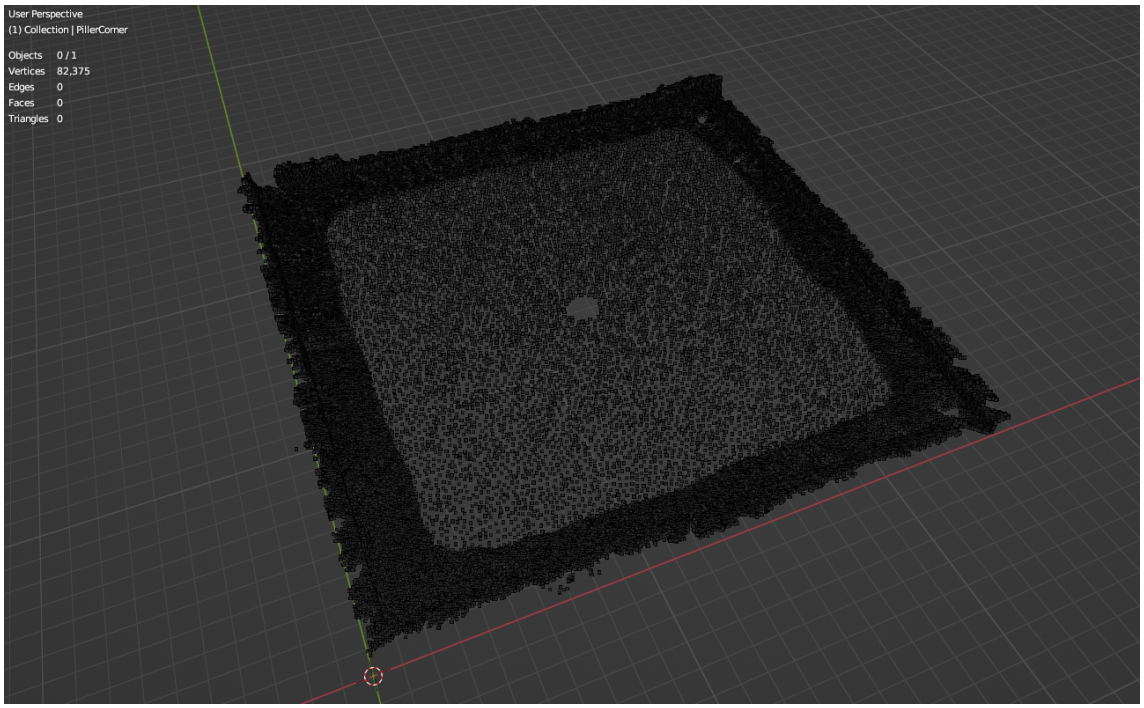
vergrabenen Elementen der Ruine wichtig. Zu diesem Zweck wird die Punktwolke grob als Fläche nachgebildet, um sie anschließend in das Unreal Engine Projekt importieren zu können. Diese Fläche dient jedoch nur als Hilfe zur Platzierung, im finalen Programm ist sie nicht sichtbar.
Allen Teilen der Visualisierung wird dann mithilfe eines Materials eine Farbe und Kontur zugewiesen. Die Farbgebung ist so nicht historisch belegt, und orientiert sich stattdessen an ähnlichen Gebäuden aus dieser Zeitperiode.

4.2.3 Zusammenführung

Nach dem Import der exportierten Modelle erstellt die Unreal Engine automatisch die benötigten Asset Dateien. Während des Imports übernimmt UE5 auch die Konvertierung von Blenders meterbasiertem Koordinatensystem hin zum dem von UE5 verwendeten, welches auf Zentimetern basiert. Die passende Größe ist wichtig, da ansonsten später die Größenverhältnisse zu den Elementen der realen Umgebung nicht mehr passen.
Die Grundstruktur des Levels bildet das Objekt, welches die virtuellen Mauern repräsentiert. Anhand dessen werden alle anderen Elemente der virtuellen Repräsentation ausgerichtet. Ein Blueprint kombiniert die Mauern mit den Elementen, die fest mit der Rekonstruktion verbunden sind. Dazu zählen die Elemente des Porticus sowie die Türen (Abbildung 4.23).
Die Türen werden durch einen separaten Blueprint repräsentiert. Innerhalb dieses Blueprints wird eine Funktionalität zum Öffnen und Schließen der Türen implementiert (Abbildung 4.24). Eine

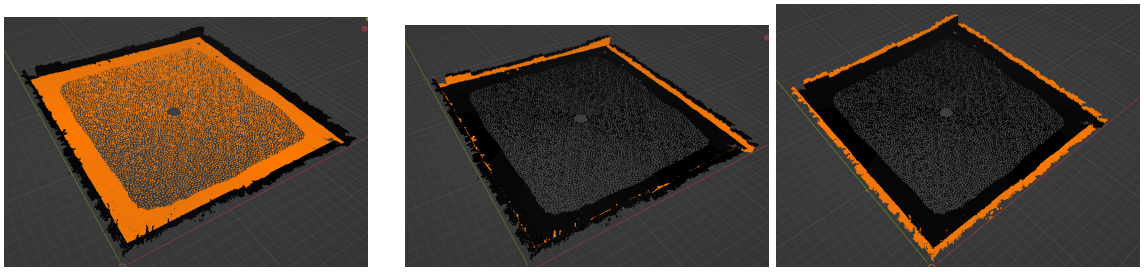


(a) Datenpunkte des Scans vor der Reduktion.



(b) Datenpunkte des Scans nach der Reduktion.

Abbildung 4.18: Visualisierung der Reduktion der Datenpunkte des Laserscans. In der Mitte werden alle Punkte innerhalb von 0.2m zusammengefügt, an den Wänden innerhalb von 0.05m.

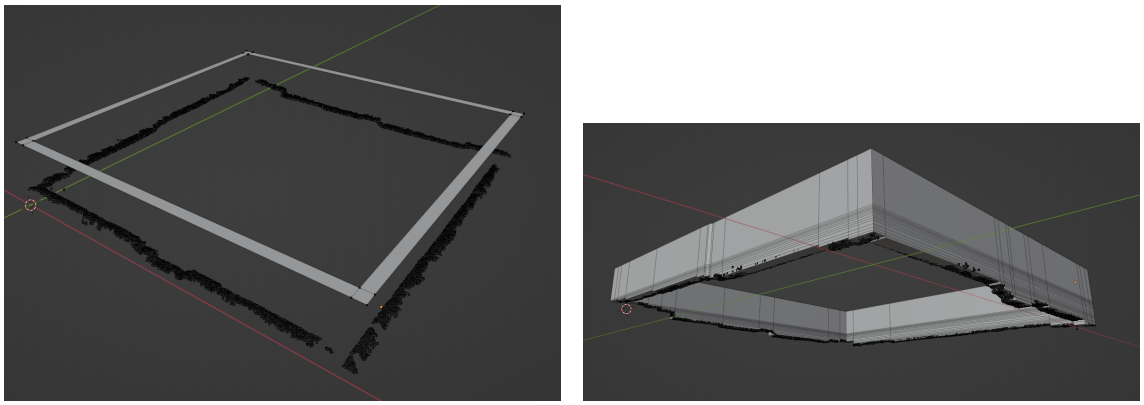


(a) Bodenfläche des Scans, notwendig für die Platzierung von unter der Erde liegenden Elementen.

(b) Seitenfläche der Wände, wichtig für die Anpassung der Rekonstruktion.

(c) Oberseite der Mauern, wichtig für die Anpassung der Rekonstruktion.

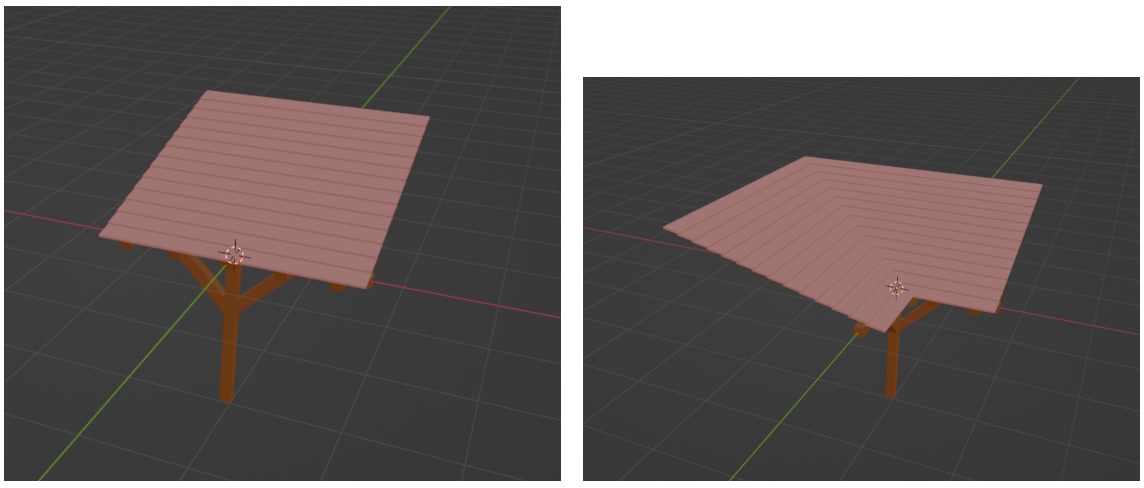
Abbildung 4.19: Darstellung der drei Unterteilungen der Punktwolke, welche die Ruine darstellt.



(a) Oberseite der realen Wand mit einer angefangenen Repräsentation.

(b) Fertige Wand, mit dem Relief der realen Ruine an der Unterseite.

Abbildung 4.20: Prozess der Erstellung der fehlenden Mauern.



(a) Rekonstruktion eines geraden Teils des Porticus.

(b) Rekonstruktion eines Eckteils des Porticus.

Abbildung 4.21: Blenderobjekte zur Rekonstruktion des Porticus.

direkte Steuerung durch den Nutzer ist nicht erforderlich. Die Tür öffnet automatisch, sobald der Nutzer sich ihr bis auf eine bestimmte Distanz nähert. Ebenso schließt sie sich wieder, wenn sich der Nutzer über eine bestimmte Schwelle hinaus von der Tür entfernt (Abbildung 4.24c). Dies wird mithilfe eines CollisionComponent im Blueprint realisiert. Dieser ist in Abbildung 4.24b als rote Kugel um die eigentliche Tür zu erkennen. Interagiert der CollisionComponent mit dem Benutzer, wird ein Event aufgerufen, welches die Türflügel rotiert und die Tür somit öffnet (Abbildung 4.24a). Ein Öffnen der Tür über das in der HoloLens integrierte Eyetracking wurde getestet, aufgrund potentiell schlechterer Benutzbarkeit der Anwendung aber wieder verworfen.
Wie im Abschnitt 4.2.2 erwähnt, wird der Porticus aus Gründen der Performance aus zwei einzelnen Bauteilen zusammengesetzt. Diese werden mehrfach in den Blueprint des Rekonstruktion eingefügt. Prinzipiell gibt es drei Möglichkeiten den Porticus mit den Außenmauern darzustellen. Im Anschluss werden diese erläutert und hinsichtlich ihrer Performance eingeordnet.
Die erste Möglichkeit ist, den Porticus in Blender als Ganzes zu erstellen und auch als Ganzes in UE5 zu importieren. Hierbei ist es auch möglich, den Porticus direkt mit den umgebenden Wänden zu verbinden, um alle konstruktiven Elemente der Szene als ein Objekt in die Szene zu stellen. Diese Umsetzung erlaubt im speziellen Fall ein Modell mit etwas weniger Punkte im Raum, da Elemente wie die Dachplatten als ein durchgehendes Mesh in das Modell integriert werden können. Zudem kann es im Editor nicht zu fehlerhaften Platzierungen oder zu unerwünschten Verschiebungen der einzelnen Elemente kommen. Ein Nachteil dieses Systems ist allerdings die fehlende Flexibilität. Jede Anpassung an der Rekonstruktion verlangt eine Anpassung des Meshes in Blender und anschließend einen Neuimport. Gerade für komplexere Strukturen ist das eine grundlegende Einschränkung. Außerdem verhindert dieses System eine wichtige Optimierung, das in UE5 sogenannte View Frustum Culling [Epi23c]. Dabei werden Objekte, welche aktuell nicht im Bereich des Sichtkegels des Nutzers liegen, automatisch verworfen, um Renderzeit einzusparen (siehe Abbildung 4.22). Aufgrund der Größe der gesamten Rekonstruktion liegen oftmals große Teile des Meshes nicht im Sichtbereich des Nutzers. Diese können aber durch das Culling nicht entfernt werden, solange ein kleiner Teil des Meshes noch sichtbar ist.
Eine Möglichkeit, um diese Probleme zu beheben, ist es, das einzelne große Mesh in mehrere Kleinere aufzuteilen. Dieser Ansatz benötigt mehr Punkte für das Mesh, hat allerdings mehrere Vorteile. Zum einen lassen sich diese kleineren Meshes beliebig miteinander kombinieren und auch mehrere Instanzen des selben Meshes unabhängig voneinander skalieren, rotieren und positionieren. Das sorgt dafür, dass bei geringfügigen Änderungen am Aussehen der Ruine oft nicht mehr das Mesh selbst in Blender angepasst werden muss. Zum anderen ermöglichen diese einzelnen Meshes der Unreal Engine, durch View Frustum Culling große Teile der Geometrie performant aus der Renderpipeline zu entfernen. Der Nachteil dieser Variante hängt mit den sogenannten Draw Calls zusammen. Diese bezeichnen den Prozess, dass die CPU Geometrie zum Rendern an die Grafikkarte überträgt. Dieser Prozess ist sehr langsam, vor allem im Vergleich mit anderen Aufgaben [Bob17]. Jedes einzelne im Level platzierte Mesh wird über einen Draw Call an die Grafikkarte übertragen. Während das einzelne große Mesh aus der vorherigen Methode mit nur einem Draw Call übertragen wird, werden für die aus den gegebenen Teilmeshes zusammengesetzte Rekonstruktion 25 benötigt. Vor allem in größeren Leveln macht dies einen erheblichen Performanceunterschied aus.
Um die Vorteile der beiden oben genannten Methoden zu vereinen, steht in UE5 ein Konzept zur Verfügung, das sich Instanced Static Mesh Component (ISMC) nennt. Dieser erlaubt das Platzieren mehrerer Instanzen desselben Meshes im Level, benötigt aber nur einen Draw Call. Hierfür wird zusätzlich zu den grundsätzlich zu übertragenden Meshdaten eine Liste mit Transformmatrizen an die GPU übergeben. Dabei beschreibt jeder Eintrag der Liste die Transformation einer Instanz des Meshes im Level. Einzelne Meshinstanzen können damit individuell platziert, rotiert und skaliert

werden (Abbildung 4.25). Der einzige Nachteil dieses Systems ist, dass auch hier, wie bereits in der vorherigen Methode, die Unreal Engine kein View Frustum Culling betreiben kann. Sobald eine Instanz des ISMC sichtbar ist, werden alle weiteren Instanzen ebenso gerendert. Diese Lösung ist dennoch effizienter als die vorher beschriebene Variante und aufgrund der Modularität besser als die erste Variante. Die Ecken und die Zwischensegmente des Porticus werden deswegen mit dieser Variante im Level platziert.
Das Anwenden dieser Technik hat vor allem Auswirkungen auf eher leistungsschwache Geräte wie die HoloLens 2. In Kombination mit weiteren Verbesserungen ist die Performancesteigerung noch besser sichtbar.
Um die layerabhängige Anzeige zu unterstützen, werden alle im Level platzierten Objekte in die drei verschiedene Gruppen Visualisierung, Information und X-Ray eingeteilt. Die Visualisierungsgruppe enthält dabei alle oberflächlich sichtbaren Objekte, die der Rekonstruktion der eigentlichen Ruine dienen. Dazu zählen die Außenmauer, der Porticus und die Türen.
Das Informationslayer beinhaltet alle platzierten Informationstafeln und zusätzlich einen in der Mitte der Ruine platzierten Richtungsweiser. Dieser befindet sich an einer fest definierten Position in der virtuellen Rekonstruktion und zeigt interessante Punkte in der Umgebung und darüber hinaus an.
Die X-Ray Ebene enthält die unter der Erde liegende Grube sowie die teilweise noch vorhandenen Fundamente des Porticus. Aufgrund eines Mangels an präzisen Daten sind diese Elemente grob anhand des Schaubildes (Abbildung 4.13 platziert und können von der realen Position abweichen.
Die für die Platzierung wichtigen Raumanker befinden sich wie in Abbildung 4.26 zu sehen an der Unterseite der Wandrekonstruktion mittig in den Ecken. Für die reale Ruine werden die selben QR-Codes wie für das Mockup verwendet (vgl. Abschnitt 4.1.2).

4.2.4 Verwendung

Die Steuerung der Anwendung funktioniert gleich wie für das Mockup, welche in Abschnitt 4.1.3 beschrieben wird. Zu Beginn muss im Hauptmenü der Button mit der Aufschrift „Ruine“ ausgewählt werden. Prinzipiell lässt sich auch dieser Level überall betrachten, aufgrund der Ausmaße der Rekonstruktion ist aber eine große Fläche ohne Hindernisse empfehlenswert. Am besten eignet sich der Standort des römisches Hauses selbst, da nur hier die Kombination realer Überreste mit der Rekonstruktion zu sehen ist. Da die HoloLens 2 bei direkter Sonneneinstrahlung aufgrund der übermäßigen Helligkeit kein perfektes Bild liefern kann, ist es außerdem empfehlenswert, einen Tag mit bewölktem Himmel auszuwählen. Wie in Abschnitt 4.2.3 beschrieben befinden sich die Raumanker zur Platzierung der Rekonstruktion an deren Ecken. Zur erstmaligen Initialisierung der Szene sollten die QR-Codes deswegen mittig auf den Ecken der Ruine platziert werden. Die genaue Anordnung ist in Abbildung 4.26 dargestellt. Wenn das Programm unabhängig der realen Ruine verwendet werden soll, ist es zu empfehlen, auf die Abstände und Höhenunterschiede der Raumanker zu achten und diese möglichst genau nachzubilden. Die genauen Maße können auch Abbildung 4.26 entnommen werden. Alternativ kann auch nur ein QR Code zur groben Platzierung verwendet werden.

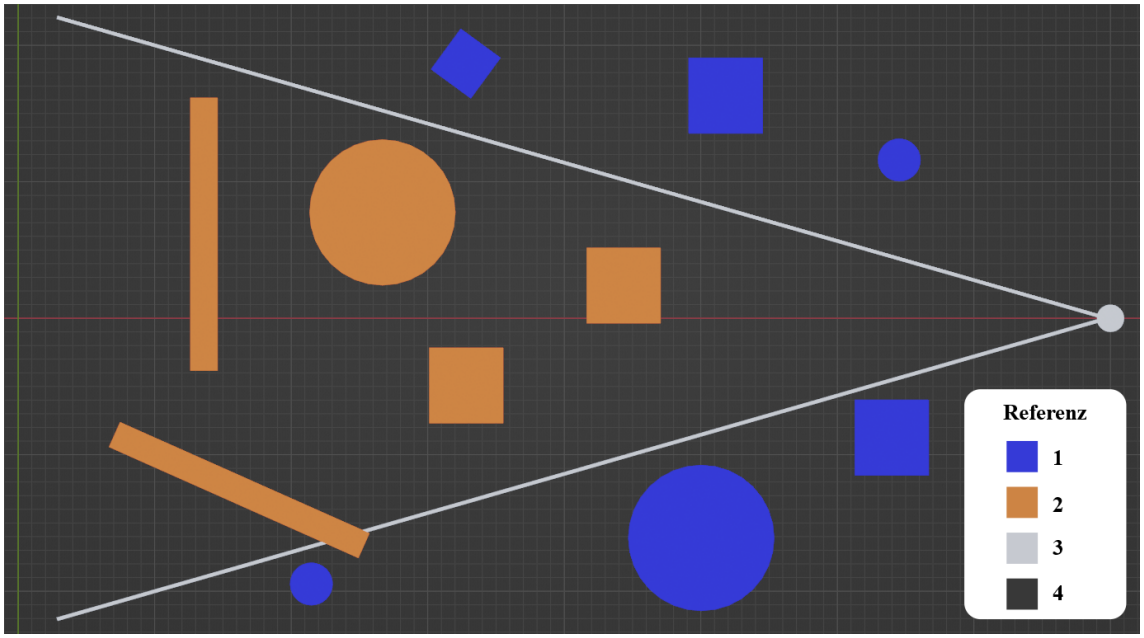


Abbildung 4.22: Auswirkungen des View Frustum Cullings. Die grauen Linien (Referenz 3) repräsentieren den Sichtkegel des Benutzers. Alles zwischen diesen Linien muss gerendert werden. Dabei werden Objekte nicht geteilt, wenn sie teilweise innerhalb und teilweise außerhalb liegen. Alle orange (Referenz 2) markierten Objekte werden gerendert, die blau (Referenz 1) Markierten werden verworfen.

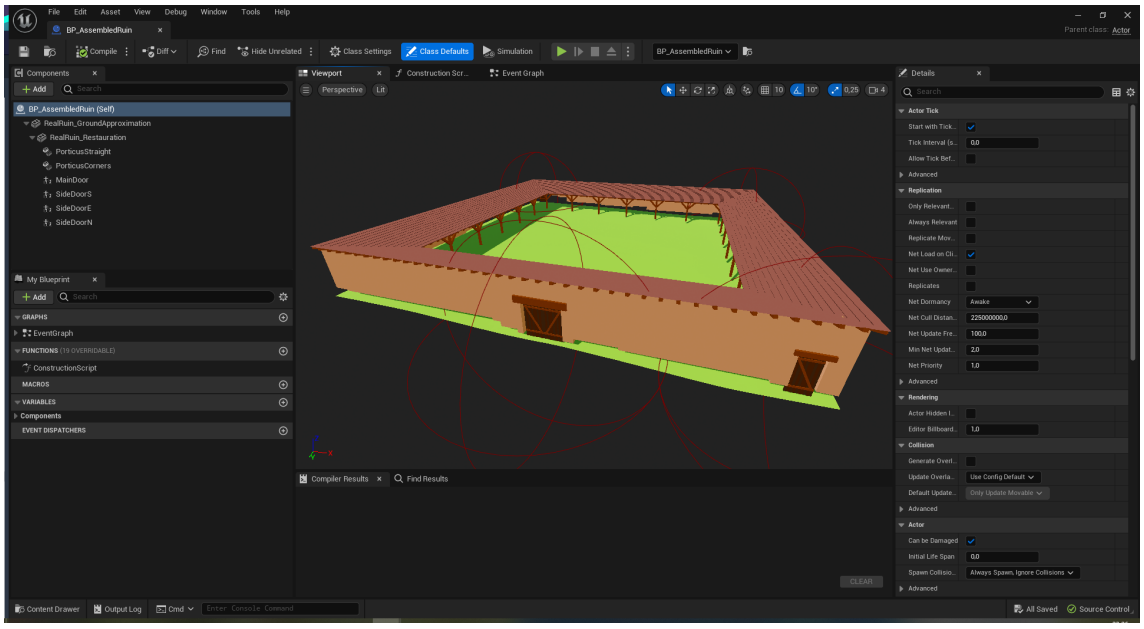
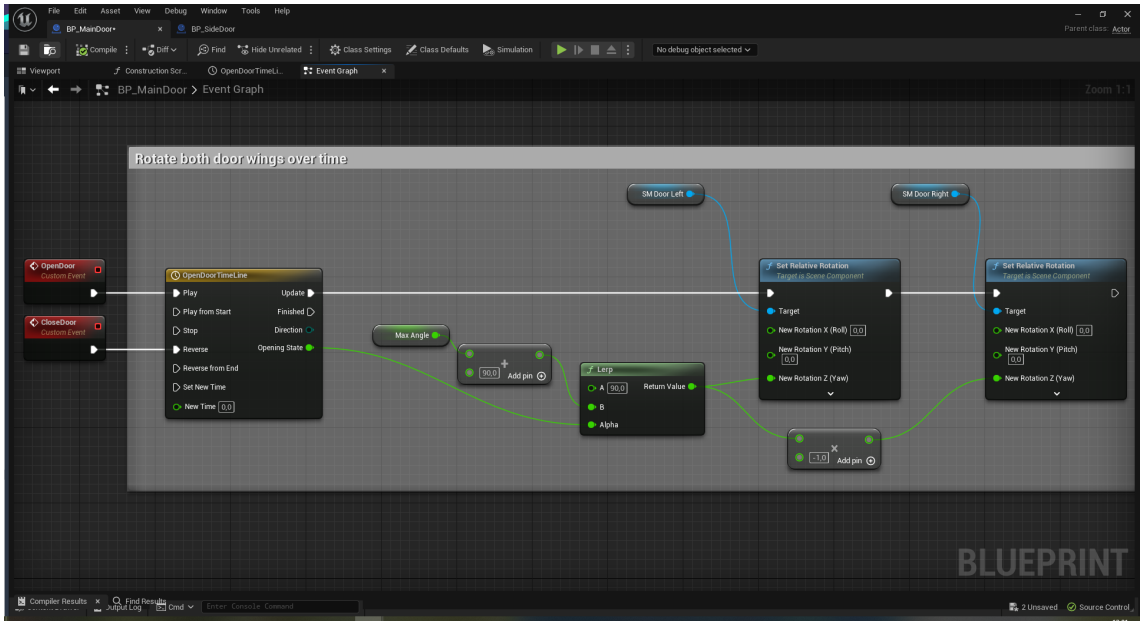
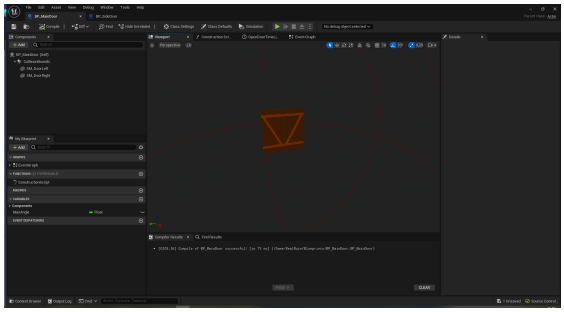


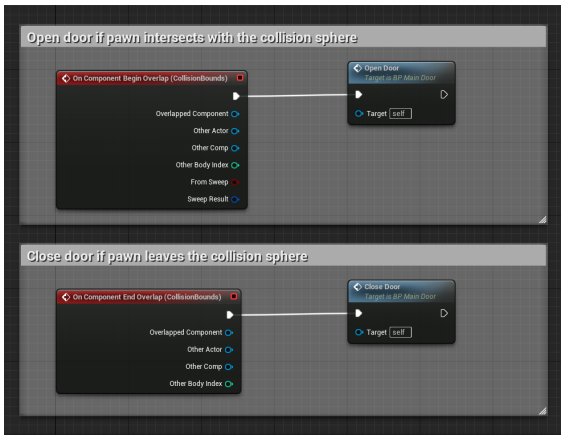
Abbildung 4.23: Ansicht des Blueprints welcher die grundlegende Rekonstruktion enthält. Die grüne Fläche ist ein Abbild des Bodens der Ruine und dient nur der Veranschaulichung, in der Visualisierung der HoloLens ist er nicht sichtbar. Die roten Kugeln repräsentieren Trigger. Wenn sich der Benutzer innerhalb dieser Trigger befindet öffnet sich die entsprechende Tür innerhalb der Visualisierung.



(a) Darstellung der Funktionalität zum Öffnen der Türen. Die dargestellte Timeline gibt über eine festgelegte Zeit anhand eines Graphen Funktionswerte aus, hier im Bereich zwischen 0 und 1. 0 steht hierbei für eine komplett geschlossene Tür und 1 für eine vollständig geöffnete. Diese Werte werden dann in eigentliche Winkel für die zwei Türflügel umgerechnet und diese dann jeweils rotiert.



(b) Darstellung der Tür mit den beiden Türflügeln und dem Bereich, in welchem sich die Tür öffnet (rote Kugel).



(c) Aufruf der Funktionen zum Öffnen und Schließen der Tür, abhängig davon ob der Benutzer den kritischen Bereich betritt oder verlässt

Abbildung 4.24: Darstellungen für die Türfunktionalität. Diese Grafiken stellen die zweiflüglige Haupttür dar. Das Projekt enthält auch noch eine Seitentür mit nur einem Türflügel. Für diese ist der Code, bis auf kleinere Anpassungen, identisch.

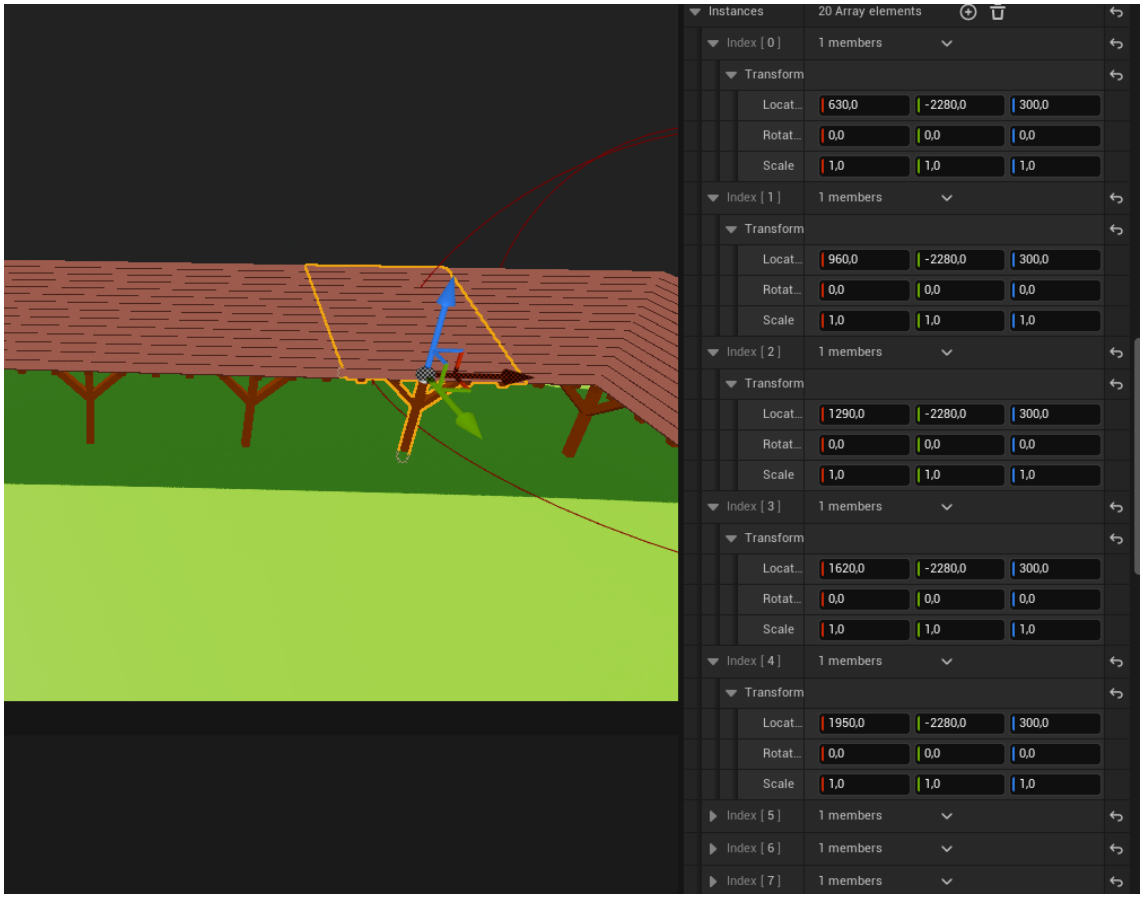


Abbildung 4.25: Ansicht des Instanced Static Mesh Component für einen Teil des Porticuses. Links ist eine Instanz hervorgehoben, rechts ist die Liste mit den verschiedenen Transformationen zu erkennen. Zu beachten ist, dass sich passend zur Position der Elemente links nur die X-Koordinaten ändern.

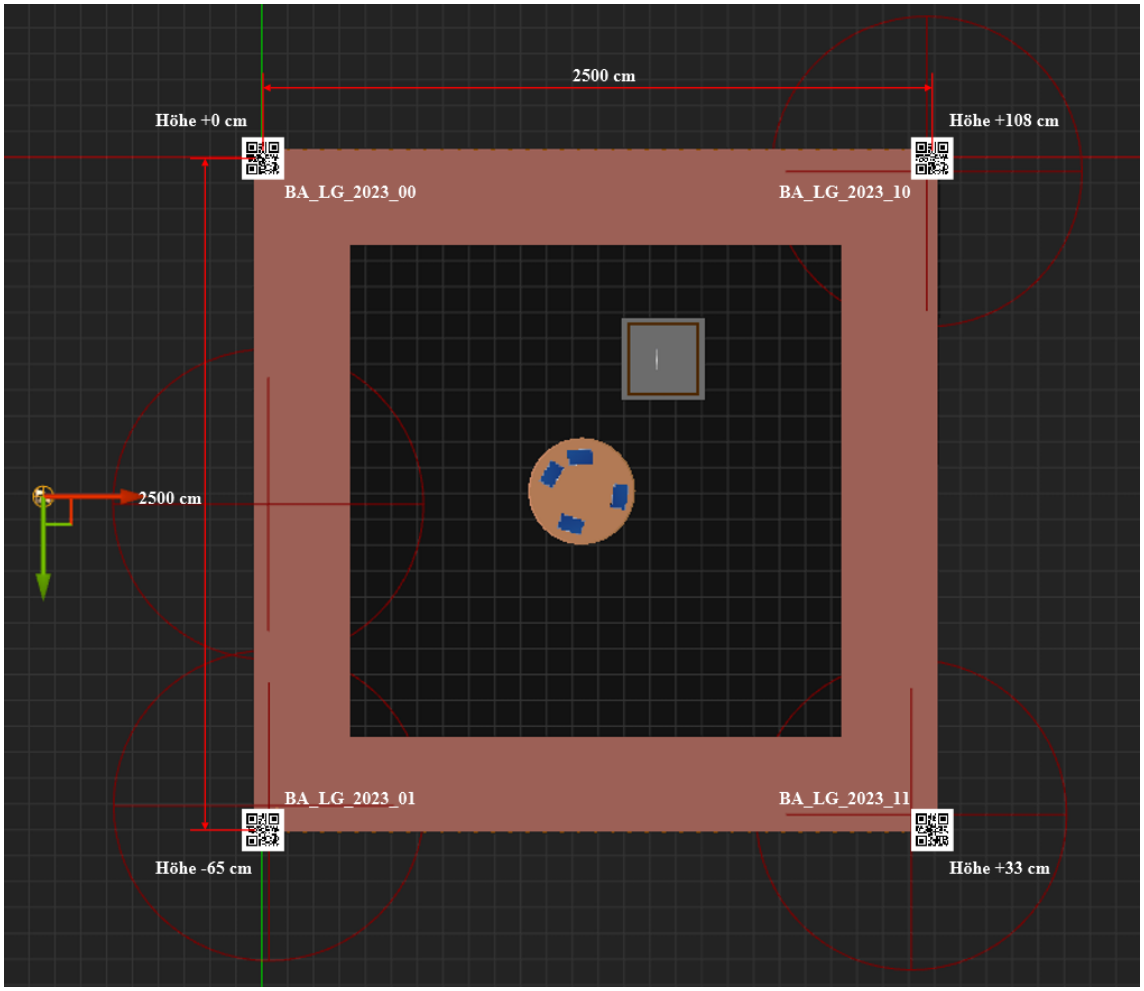


Abbildung 4.26: Ansicht der Platzierung der QR-Codes an der Ruine. Zu beachten ist, dass sich die QR Codes auch auf unterschiedlichen Höhen befinden müssen. Diese Höhen sind gegeben durch die Höhendaten in der Grafik. Diese sind relativ zu der Höhe des QR-Codes mit dem Wert BA_LG_2023_00 gegeben. Bei Verwendung vor Ort am römischen Haus stimmen die Oberseiten der Mauern mit diesen Maßen überein.

5 Diskussion

Dieses Kapitel behandelt das Ergebnis der Arbeit, vor allem bezogen auf die Rekonstruktion der Ruine. Für Informationen über das Mockup siehe Abschnitt 4.1. Weiterhin werden Probleme im Ergebnis diskutiert, der Übertrag der Funktionalitäten auf andere Ruinen erläutert und weitere Anwendungsgebiete vorgestellt.

5.1 Ergebnis

Dieser Teil behandelt die Kombination der realen Ruinen des römischen Hauses mit der Rekonstruktion. Alle referenzierten Bilder zeigen Screenshots, die mit der HoloLens 2 angefertigt wurden. Abbildung 5.1 liefert einen Überblick über die verschiedenen Sichtebenen der Anwendung. Alle wichtigen Teile der Rekonstruktion sind in diesen Fotos dargestellt. Zusätzlich zeigen Abbildung 5.2 und Abbildung 5.3 zwei Besonderheiten der Rekonstruktion. Ersteres zeigt die interaktiven Türen, welche sich wie in Abschnitt 4.2.3 öffnen, sobald sich der Benutzer nähert. Abbildung 5.3 zeigt den virtuellen Kompass. Dieser gibt Informationen über interessante Orte in der Umgebung. Zuletzt stellt Abbildung 5.4 noch die Genauigkeit der Rekonstruktion im Bezug auf die Kontur der Mauerkante dar. Durch den Laserscan, beschrieben in Abschnitt 4.2.1, können zentimetergenau die unterschiedlichen Mauerstufen abgebildet werden.
Das Problem, welches auch schon für die verschobenen Scans der HoloLens (Abschnitt 4.2.1) verantwortlich war, zeigt sich auch bei der Kombination der Rekonstruktion mit der realen Ruine. Die Platzierung über die QR-Codes funktioniert und ist im Vorfeld der Screenshots auch mehrmals durchgeführt worden. Nach größeren Bewegungen fehlt der HoloLens allerdings die Präzision im Tracking, was sich durch die verschobene Rekonstruktion bemerkbar macht. Besonders deutlich zu erkennen ist dies in Abbildung 5.1b. In diesem Fall ist es vor allem eine Verschiebung in der Höhe. Abweichung von bis zu einem Meter in horizontaler Richtung kommen jedoch auch vor. Die in Abschnitt 4.1.2 und Abschnitt 5.5.1, kombiniert mit dem GPS-Tracking aus Abschnitt 5.5.4, kann dieses Problem durch kontinuierliches Aktualisieren jedoch lösen. Auch wenn die Verschiebungen die Illusion, im Gebäude zu stehen, teilweise verschlechtert, so stellt die Arbeit doch eine sinnvolle Möglichkeit dar, Ruinen interaktiv zu rekonstruieren.

5.2 Übertrag auf andere Ruinen

Die für diese Arbeit verwendete Methodik zur Erstellung einer Rekonstruktion lässt sich für viele Anwendungsfälle ohne technische Anpassungen auf andere Ruinen übertragen. Wichtig sind eine genaue Vermessung der zu rekonstruierenden Ruine, beispielsweise wie in dieser Arbeit mit Hilfe eines Laserscanners. Im Anschluss kann dann die eigentliche Rekonstruktion erstellt werden. Dabei ist es vor allem bei komplexeren Rekonstruktionen nötig, auf Hardware- und

(a) Reine reale Ansicht ohne Overlay.

(b) Ansicht mit aktivierter Visualisierungsebene.

(c) Ansicht mit aktivierter Visualisierungs- und Informationsebene.

(d) Ansicht mit aktivierter Visualisierungs-, Informations- und X-Rayebene.

Abbildung 5.1: Ansicht eines Teils der Ruine des römischen Hauses, mit verschiedenen aktivierten Overlays.

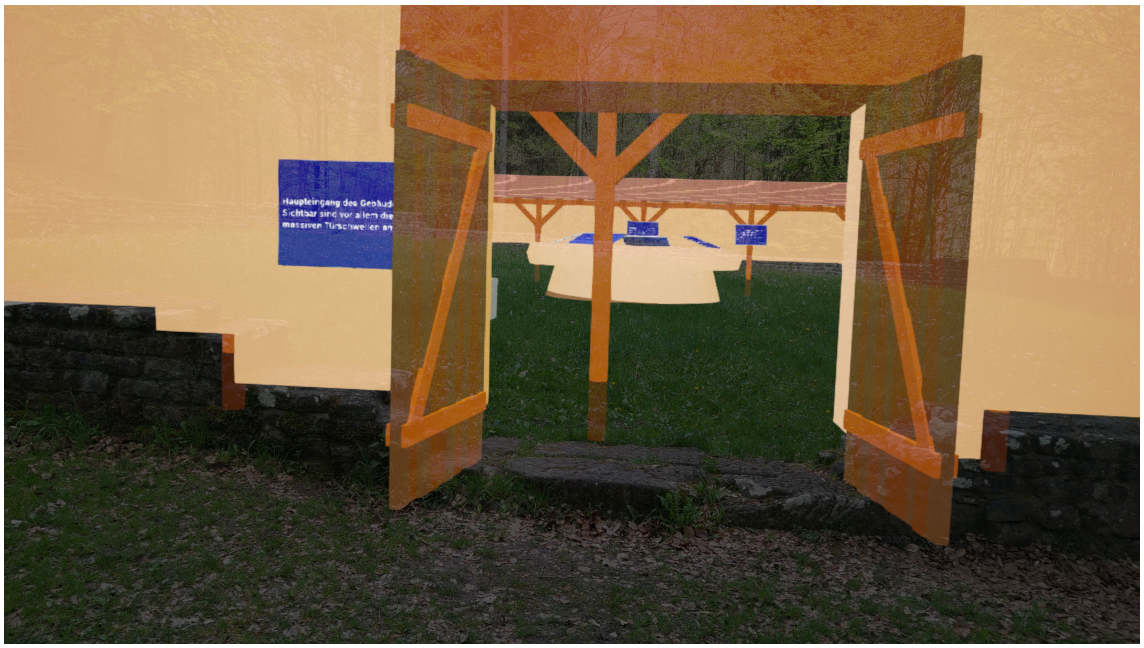


Abbildung 5.2: Ansicht des rekonstruierten Haupteingangs des römischen Hauses. Links und rechts bei den Balken ist das in Abschnitt 5.4.2 beschriebene Problem mit der falschen Überdeckung zu erkennen.

Abbildung 5.3: Ansicht des Kompasses, welcher Informationen über naheliegende Orte anzeigt

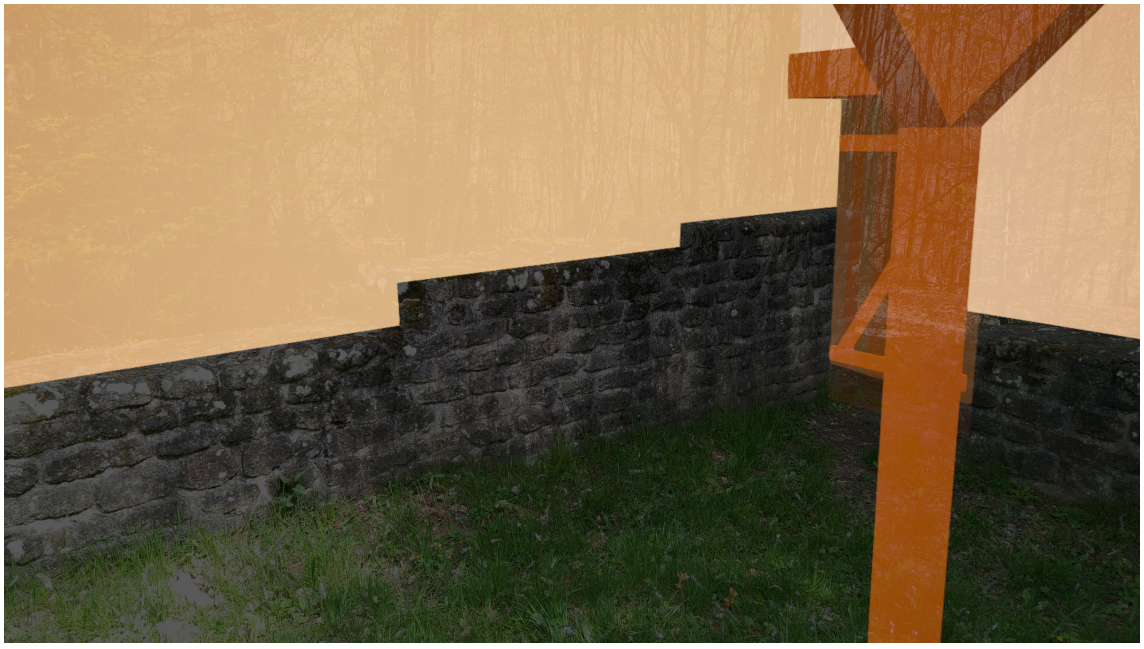


Abbildung 5.4: Beispielansicht, wie genau die Rekonstruktion mit der realen Ruine zusammenpasst

Performanceeinschränkungen zu achten und die in Abschnitt 4.2.2 benannten Optimierungen umzusetzen. Die Platzierung der Rekonstruktion erfolgt dann ebenso über QR-Code basierte Raumanker. Je nach Architektur kann es nötig sein, mehr als vier Raumanker zu verwenden. Außerdem kann es je nach Layout der Ruine vorkommen, dass die in Abschnitt 5.4.2 beschriebene Situation eintritt. Dann müssen für ein realistisches Ergebnis die dort beschriebenen Lösungen umgesetzt werden.

5.3 Anwendungen

Unsere Arbeit kann für verschiedene Anwendungsgebiete verwendet werden. Vor allem der Tourismus sowie wissenschaftliche Bereiche können von solchen Visualisierungen profitieren. Für Touristen bietet diese Technologie eine neue und interessante Möglichkeit, Geschichte zu erleben [FDC+19] [HLW09]. Im Tourismussektor scheint es auch sinnvoll, das in Abschnitt 5.4.1 beschriebene Levelstreaming zu implementieren. Im Bereich der Archäologie kann es von Nutzen sein, noch im Boden verborgene Bereiche der Ausgrabungsstätte bereits im Voraus darzustellen. Solche Daten können beispielsweise durch sogenanntes Bodenradar [S Z] gewonnen werden. Visualisierungen basierend auf diesen Daten können helfen, die Ausgrabungen effizienter durchzuführen, und Beschädigungen während der Ausgrabung durch verwendete Geräte zu vermeiden.

5.4 Limitierungen

Dieses Kapitel beschäftigt sich mit Problemen, die verschiedene weitere Features der Arbeit beschränken oder verhindern.

5.4.1

Leistung

Viele der bereits benannten Einschränkungen basieren auf der Leistungsfähigkeit der integrierten Hardware der HoloLens 2. Eine Anwendung hochaufgelöster Texturen ist aufgrund des beschränkten Grafikspeichers nur sehr eingeschränkt möglich, und auch die Komplexität der Meshes sollte möglichst gering gehalten werden. Potentielle Lösungen für diese Probleme sind unter anderem portable Computer oder auch Datenstreaming über ein W-Lan Netzwerk. In beiden Fällen wird die HoloLens nur noch zur Nutzereingabe und als Display verwendet. Die aufwändige Rechenarbeit übernehmen die externen Computer. Ein Nachteil des portablen Computers ist dessen großes Gewicht und die dadurch eingeschränkte Mobilität des Nutzers. Diese Beschränkungen hat die Lösung des Datenstreamings nicht. Für den Anwender gibt es keine Änderungen, da die Computer zur Berechnung des Bildes extern stehen können. Allerdings ist hierfür je nach Qualität des gerenderten Bildes eine sehr schnelle Datenverbindung nötig. Außerdem kann bei diesem System der sogenannte Input-Lag problematisch werden. Dieser beschreibt die Verzögerung zwischen einer Eingabe des Nutzers und dem Darstellen des jeweiligen Bildes auf der Brille. Da insbesondere Kopfbewegungen sehr schnell ausgeführt werden können, muss sichergestellt sein, dass die Bilder rechtzeitig auf der Brille angezeigt werden. Ansonsten kann das zu einem Bild führen, welches

nicht mehr zu der aktuellen Ausrichtung des Anwenders passt. Dieses Problem lässt sich für gewisse Benutzereingaben lösen, indem sie direkt auf der Brille verarbeitet werden. Das erfordert jedoch technische Änderungen, welche aktuell nicht in der Arbeit implementiert sind.

5.4.2 Überlappung

Eine aktuell nicht berücksichtigte Einschränkung stellt das korrekte Verdecken von virtuell rekonstruierten Ruinenteilen durch reale Objekte dar. Da immer der komplette virtuelle Teil über das reale Bild gerendert wird kann es vorkommen, dass virtuelle Wände, die sich eigentlich hinter einer realen Säule befinden sollten, für den Benutzer vor der Säule angezeigt werden. Dieses Problem kann beispielsweise im Mockup gesehen werden. Unabhängig davon, ob das Gebäude in den realen Raum passt oder nicht, wird es immer vor allen realen Objekten gerendert. Eine Lösung dafür bietet das Spatial Meshes der HoloLens. Hierfür werden Teile der virtuellen Ansicht entfernt, wenn sich das Spatial Mesh vor diesen befindet. Da ein solcher Fall für die Ruine des römischen Hauses nicht störend auftritt, ist dieses Feature in der aktuellen Ausführung der Arbeit nicht enthalten. In Abbildung 5.2 ist ein solcher Fall dennoch bei den Holzbalken zu erkennen.

5.4.3 Farben

Eine für diese Arbeit wichtige Eigenschaft der HoloLens 2 stellt die verwendete Renderingmethode dar. Über die Ansicht der realen Welt wird ein Glas gelegt, auf welches Hologramme projiziert werden. Das sorgt dafür, dass die Farben der Hologramme immer heller sein sollten als der Hintergrund. Die Brille fügt Farben dem Bild der realen Welt hinzu und kann keine entfernen. So sind Hologramme vor einem schwarzen Hintergrund sehr gut zu sehen, vor einem Weißen allerdings nur sehr schwer.

5.5 Mögliche Erweiterungen

Dieses Kapitel beschreibt mögliche weitergehende Features. Einige wurden bereits im Laufe dieser Arbeit genannt, andere nehmen die Ideen der Arbeit auf und beschreiben diese ausführlicher.

5.5.1 Platzierung über Heightmap und Image Matching

Die in Abschnitt 4.1.2 benannte Methode zur Platzierung der virtuellen Elemente über Vergleiche von Höhenkarten ist weiterhin eine sinnvolle Option. Dieses Feature ist aufgrund von Performanceproblemen nicht im Endstand dieser Arbeit enthalten. Durch effizientere Algorithmen und den Fortschritt der Technik in den nächsten Jahren ist es denkbar, diese Methode dann in das Projekt zu integrieren.

5.5.2 Umgebungsgestaltung

Eine Möglichkeit, die Immersion zu erhöhen, ist das Einbinden von weiteren Elementen in die Szene. So ist es denkbar, zur damaligen Zeit passende Vegetation im Hintergrund anzuzeigen, oder weitere Dekorationen nach damaligem Vorbild in der Szene zu platzieren. Je nach Ruine und dazugehörigen Aufzeichnungen kann das zu unterschiedlichen Detailgraden der Realität entsprechen. Für das römische Haus existieren solche Aufzeichnungen nicht, weswegen das Einbinden dieser Detailtiefe im Projekt nicht umgesetzt wird. Darauf aufbauend ist es auch denkbar, rekonstruierte Ruinen mit zeitgenössischem Leben zu füllen. Vor allem Menschen und Tieren bieten sich hierfür an. Das in Kapitel 3 beschriebene Archeoguide enthält bereits ein ähnliches Feature [VKT+01]. Je nach Ruine lassen sich dadurch sehr gut ehemalige Funktionen und Anwendungsgebiete von verschiedenen Bauten visualisieren. Gerade diese Art der Erweiterung ist allerdings in vielen Bereichen sehr aufwendig und deswegen im aktuellen Projekt nicht umgesetzt. Vor allem aufgrund der Performance kann es für dieses Feature sehr sinnvoll sein, zusätzlich die in Abschnitt 5.4.1 beschriebene externe Bildberechnung zu implementieren.

5.5.3 Platzieren vorhandener Teile

Eine andere interessante Erweiterung ist das Einbinden von realen Objekten, die sich nicht mehr an ihrer ursprünglichen Position befinden. Das kann dem Benutzer weiterhin dabei helfen, die aktuellen Gegebenheiten im historischen Kontext korrekt einzuordnen. Eine Möglichkeit der Umsetzung ist es, die realen Überreste mit einer Umrandung hervorzuheben und eine Umrandung an dem Ort zu platzieren, an dem sich das Objekt ursprünglich befand. Da solche Elemente beim römischen Haus nicht vorhanden sind, konnte dies nicht in diese Arbeit eingebunden werden.

5.5.4 GPS-unterstützte Lokalisierung

Wie in Abschnitt 4.1.2 bereits beschrieben kann es gerade bei größeren Gebieten zum Orientierungsverlust der HoloLens kommen. Dieses Problem tritt vor allem in Bereichen auf, in denen wenige eindeutige Landschaftsmarker vorhanden sind. Beim römischen Haus sind das beispielsweise die freie Wiesenfläche in der Mitte der Ruine sowie die sehr ähnlich aussehenden Mauerreste. Um diese Probleme zu beheben, ist es denkbar, das integrierte Lokalisierungssystem der HoloLens über GPS zu unterstützen. Dabei liefert das GPS eine grobe Position, und die integrierte räumliche Erkennung der HoloLens übernimmt die exakte Platzierung. Um die Genauigkeit der GPS-Position zu verbessern, bietet sich die Nutzung von fest definierten GPS-Markern an. Wenn die GPS-Position des Nutzers in Relation zu der des Markers gesetzt wird, ist es möglich, Ungenauigkeiten zu eliminieren. Diese Technik wird beispielsweise von Archeoguide bereits verwendet [VKT+01].

6 Zusammenfassung und Ausblick

Unsere Arbeit behandelt die Rekonstruktion von Ruinen mithilfe von Augmented Reality. Das Ziel ist, Geschichte durch das Einbinden moderner Techniken erlebbar zu machen. Der Fokus liegt hierbei auf einer einfachen Benutzbarkeit bei gleichzeitig hoher Informationsdichte. Außerdem steht der Spaß im Vordergrund, mit dem Besucher einer Ruinenstätte die Geschichten im Hintergrund entdecken können.
Als Beispielruine wird das römische Haus in Stuttgart gewählt, da sich die Komplexität und Erreichbarkeit des Ortes perfekt für einen Prototypen der Rekonstruktionsanwendung eignet. Die Rekonstruktion selbst wird grob aufgrund von offiziellen Vermutungungen zum ursprünglichen Aussehen erstellt. Unterstützt durch moderne Technik wie Laserscanner und eine Game-Engine wird dann die Ruine virtuell wieder aufgebaut und im Anschluss mithilfe der HoloLens 2 mit der realen Ruine zu einem Ganzen kombiniert. Die Kombination der Unreal Engine mit der HoloLens 2 eignet sich für die gegebene Aufgabe sehr gut. Das liegt hauptsächlich an der guten Integration von AR spezifischen Plugins in UE5, aber auch an der für diese Art der Anwendung gut geeigneten Hardware der HoloLens 2.

6.1 Ausblick

Die Platzierung der Rekonstruktion mithilfe von GPS-Lokalisierung und Heightmap Image Matching, wie beschrieben in Abschnitt 5.5.4 und Abschnitt 5.5.1, ist der nächste Schritt in der Verbesserung des Projektes. Mit der Implementierung dieser Funktion wird das größte aktuell vorhandene Problem, beschrieben in Abschnitt 5.1, behoben. Detailliertere und umfangreichere Visualisierungen können dann darüber hinaus die Simulationstiefe der Anwendung noch weiter steigern.
Mit der steigenden Leistungsfähigkeit von Kleincomputern wird sich in den nächsten Jahren die AR-Technologie immer weiter entwickeln. Programme wie die in dieser Arbeit beschrieben Visualisierung profitieren stark von dieser Entwicklung. Durch vergünstigte Anschaffungskosten lassen sich auch touristisch immer mehr Bereiche und Ruinen erschließen.

Literaturverzeichnis

[21] *Villa Rustica*. 2021. URL: https://www.roemervilla.de/villa-rustica (zitiert auf S. 34).

[BDPC12] A. Bernardini, C. Delogu, E. Pallotti, L. Costantini. „Living the Past: Augmented Reality and Archeology“. In: Aug. 2012. DOI: 10.1109/ICMEW.2012.67 (zitiert auf S. 17, 19).

[Ben15] Benjamin Layug. *Ten things you may not know about Corregidor*. 10.05.2015. URL: https://businessmirror.com.ph/2015/05/10/ten-things-you-may-not-know-about-corregidor/ (zitiert auf S. 18).

[Ble23] Blender foundation. *Blender*. 2023. URL: https://www.blender.org/ (zitiert auf S. 14, 15).

[Bob17] Bob Cober. *UE4 - Overview of Static Mesh Optimization Options*. 7.01.2017. URL: https://www.casualdistractiongames.com/post/2017/01/07/ue4-overview-of-static-mesh-optimization-options (zitiert auf S. 41).

[Epi23a] Epic Games. 2023. URL: https://www.epicgames.com/site/de/home?sessionInvalidated=true (zitiert auf S. 14).

[Epi23b] Epic Games. *Unreal Engine 5*. 2023. URL: https://www.unrealengine.com/en-US/unreal-engine-5 (zitiert auf S. 14).

[Epi23c] Epic Games. *Visibility and Occlusion Culling: An overview of available visibility and occlusion culling methods.* 2004-2023. URL: https://docs.unrealengine.com/5.1/en-US/visibility-and-occlusion-culling-in-unreal-engine/ (zitiert auf S. 41).

[FDC+19] N. Flores, L. I. Dolores, G. Cayabyab, T. Palaoag, J. Angeles, G. Corpuz, R. Samson, J. Dela Cruz, M. Mamaril. „Rebuilding The Corregidor Ruins In An Augmented Reality Environment and Its Usability“. In: Okt. 2019 (zitiert auf S. 18, 19, 50).

[Gut10] D. A. Guttentag. „Virtual reality: Applications and implications for tourism“. In: *Tourism Management* 31.5 (2010), S. 637–651. ISSN: 0261-5177. DOI: https://doi.org/10.1016/j.tourman.2009.07.003. URL: https://www.sciencedirect.com/science/article/pii/S0261517709001332 (zitiert auf S. 19).

[Hei] Heidelberger Kunstverein. *Panorama*. URL: https://hdkv.de/leseraum/panorama/ (zitiert auf S. 17).

[HLW09] Y. Huang, Y. Liu, Y. Wang. „AR-View: An augmented reality device for digital reconstruction of Yuangmingyuan“. In: *2009 IEEE International Symposium on Mixed and Augmented Reality - Arts, Media and Humanities*. 2009, S. 3–7. DOI: 10.1109/ISMAR-AMH.2009.5336752 (zitiert auf S. 19, 50).

[Khr23] Khronos Group. *OpenXR standard*. 2023. URL: https://www.khronos.org/api/index_2017/openxr (zitiert auf S. 14).

[KR10] J. Kang, J.-h. Ryu. „Augmented Reality Window: Digital reconstruction of a historical and cultural site for smart phones“. In: *2010 IEEE International Symposium on Mixed and Augmented Reality - Arts, Media, and Humanities*. 2010, S. 67–68. DOI: 10.1109/ISMAR-AMH.2010.5643289 (zitiert auf S. 17, 19).

[LE21] L. J. Liang, S. Elliot. „A systematic review of augmented reality tourism research: What is now and what is next?“ In: *Tourism and Hospitality Research* 21.1 (2021), S. 15–30. DOI: 10.1177/1467358420941913. eprint: https://doi.org/10.1177/1467358420941913. URL: https://doi.org/10.1177/1467358420941913 (zitiert auf S. 19).

[Lei23] Leica Geosystems AG - Part of Hexagon. *Leica BLK360 Imaging Laser Scanner*. 2023. URL: https://leica-geosystems.com/products/laser-scanners/scanners/blk360 (zitiert auf S. 13).

[Mic] Microsoft. *HoloLens 2*. URL: https://www.microsoft.com/de-de/d/hololens-2/91pnzzznzwcp?activetab=pivot:technischendatentab#tab12f4856b4-8345-4aa8-af70-10cd83bca050 (zitiert auf S. 18).

[Mic22a] Microsoft. 2022. URL: https://github.com/microsoft/Microsoft-OpenXR-Unreal (zitiert auf S. 14).

[Mic22b] Microsoft. 2022. URL: https://github.com/microsoft/MixedReality-UXTools-Unreal (zitiert auf S. 14).

[Mic22c] Microsoft. 2022. URL: https://github.com/microsoft/WorldLockingTools-Unreal (zitiert auf S. 14).

[Mic23] Microsoft. *Microsoft Hololens 2*. 2023. URL: https://www.microsoft.com/de-de/hololens/ (zitiert auf S. 13).

[Nic] Nick Mower. *A Practical Guide to Unreal Engine 4's Coordinate System*. URL: https://www.techarthub.com/a-practical-guide-to-unreal-engine-4s-coordinate-system/ (zitiert auf S. 22).

[Pla05] D. Planck. „Die Römer in Baden-Württemberg - Römerstätten und Museen von Aalen bis Zwiefalten“. In: Stuttgart: Theiss, 2005. Kap. Stuttgart: Römisches Haus im Rotwildpark. 326f. ISBN: 978-3-806-21555-7 (zitiert auf S. 15, 34).

[Qua] Qualcomm. *Snapdragon 850 Mobile Compute Platform*. URL: https://www.qualcomm.com/products/mobile/snapdragon/pcs-and-tablets/snapdragon-8-series-mobile-compute-platforms/snapdragon-850-mobile-compute-platform (zitiert auf S. 18).

[S Z] S. Zickgraf. *Bodenradaruntersuchung in der Archäologie: Erkundung unterirdischer massiver Baubefunde, Mauern, Fundamente und Schuttansammlungen*. URL: https://www.pzp.de/geophysikalische-untersuchungsmethoden-in-der-archaeologie/bodenradar-untersuchung-in-der-archaeologie.html (zitiert auf S. 50).

[SB11] G. Saggio, D. Borra. „Augmented Reality for Restoration/Reconstruction of Artefacts with Artistic or Historical Value“. In: Dez. 2011. ISBN: 978-953-307-422-1. DOI: 10.5772/27066 (zitiert auf S. 17).

[the23] thetuvix, vtieto, v-hearya and DCtheGeek. *Koordinatensysteme*. 21.03.2023. URL: https://learn.microsoft.com/de-de/windows/mixed-reality/design/coordinate-systems (zitiert auf S. 22).

[Uni23] Unity. 2023. URL: https://unity.com/de (zitiert auf S. 14, 18).

[Var23] Varjo. *Varjo XR-3*. 2023. URL: https://varjo.com/products/xr-3/ (zitiert auf S. 14).

[VKT+01] V. Vlahakis, J. Karigiannis, M. Tsotros, M. Gounaris, L. Almeida, D. Stricker, T. Gleue, I. Christou, N. Ioannidis. „ARCHEOGUIDE: first results of an augmented reality, mobile computing system in cultural heritage sites“. In: Jan. 2001, S. 131–140. DOI: 10.1145/584993.585015 (zitiert auf S. 18, 52).

Alle URLs wurden zuletzt am 15. 05. 2023 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Druck-Exemplaren überein.

Ort, Datum, Unterschrift