

InfoStructure VR

Dokumentation

Projektmitglieder

Julia Danko

Kevin Broy

Kevin Drabinski

Bernd Adamczyk

Vaceslav Klepšov

Projektbetreuer

Prof. Dr. Gregor Lux

Dipl. Ing. Thomas Kollakowsky

Abgabedatum

18. März 2020

Inhaltsverzeichnis

Pflichtenheft	7
Lastenheft.....	7
Produktbeschreibung	7
Projektdurchführung	8
Methodik	8
Werkzeuge.....	9
Entwicklung	9
Design	11
Organisation	12
Kommunikation	13
Dokumentation	14
Produktbedienung.....	16
Videoanleitung:	16
Steuerung	16
Tooltip.....	16
Menü.....	16
VR	17
Auswählen.....	17
Teleportieren	17
Isolieren	17
X-Ray	17
Zurücksetzen:.....	18
Während des „Transform-Modus“	18
Alles Zurücksetzen	18
Menü	18
Steam Menü.....	18
Zoom.....	19

Metadaten	19
Zurück zum Start.....	19
Transform-Modus.....	19
Objekt bewegen	20
Objekt wegschieben	20
Objekt heranziehen.....	20
Desktop	22
Benutzermodi	22
Benutzermodus wechseln	22
Allgemein	22
Shortcuts für Menü-Tools.....	23
Transform-Modus	23
Projektaufbau.....	24
Klassen der Benutzerfigur	24
Metadaten aus Excel (Java).....	24
Metadaten aus IFCXML (Java)	25
User Interface Aufbau	25
Abweichungen vom Pflichtenheft	28
Nicht erfüllte Pflichtenheft-Anforderungen.....	28
Musskriterien.....	28
Wunschkriterien.....	28
Konfigurationen	29
Multi-User.....	29
Tools.....	29
Ergänzungen zum Pflichtenheft	29
Designentscheidungen	30
Collab Viewer Template	30
Menü	30

Technische Entscheidungen.....	32
Wahl der Programmiersprache.....	32
Wahl des Dateiformats für die Metadaten.....	32
VR-Brillen.....	32
Beleuchtung.....	32
Aufbau des Projekts.....	33
Implementierung des Funktionsumfangs.....	33
Einbindung eines Voice over IP-Dienstes (VoIP).....	33
Technische Herausforderungen.....	34
Metadaten der IFC.....	34
Zoom in der VR.....	37
Oculus Quest.....	45
Automatische Texturierung.....	46
Verantwortlichkeiten.....	48
Gruppe UE4 (Julia Danko, Kevin Drabinski, Vaceslav Klepzov).....	48
Gruppe IFC (Kevin Broy, Bernd Adamczyk).....	48
Bernd Adamczyk.....	48
Kevin Broy.....	49
Julia Danko.....	49
Kevin Drabinski.....	50
Vaceslav Klepzov.....	50
Logbuch.....	51
Videos.....	51
Referenzen und Literatur.....	51
Literatur:.....	51
Referenzen:.....	51
Weiteres.....	52
Projektmanagement Präsentation.....	52
Show & Tell Plakat.....	52

Abkürzungsverzeichnis

Abb	Abbildung
BIM	Building Information Modeling
BP	Blueprint
BPs	Blueprints
CSS	Cascading Style Sheets
CSV	Comma-Separated Values
HTML	Hypertext Markup Language
IFC	Industry Foundation Class
JSON	JavaScript Object Notation
S	Siehe
UE	Unreal Engine
UI	User Interface
VR	Virtual Reality
VoIP	Voice over IP
WMR	Windows Mixed Reality
z.B.	Zum Beispiel

Abbildungsverzeichnis

Abbildung 1: Klassen der Benutzerfigur im Projekt.....	24
Abbildung 2: Metadaten aus Excel	24
Abbildung 3: Metadaten aus IFCXML.....	25
Abbildung 4: Zustandsdiagramm für das Menü.....	26
Abbildung 5: Zustandsdiagramm für die Metadatenanzeige.....	27
Abbildung 6: Sequenzdiagramm für die Metadatenanzeige	27

Abbildung 7: Controller mit dem alten Menü	30
Abbildung 8: Controller mit dem Tooltip	31
Abbildung 9: Controller mit dem neuen Menü	31
Abbildung 10: Extrahierte Metadaten aus einem IFCXML-Dokument.....	35
Abbildung 11: Ausgegebene JSON-Datei in einem Viewer	36
Abbildung 12: Metadaten aus einem Excel-Dokument als CSV-Datei	37
Abbildung 13: Kamera im Motion Controller Blueprint abhängig vom Motion Controller	37
Abbildung 14: Anzeige des Zoom Widgets mit schlechter Qualität	38
Abbildung 15: Lösung: Custom Depth Stencil White Mask: All bits (255), ignore depth.....	39
Abbildung 16: Geöffnete Lupe. Der ausgewählte Rahmen erscheint mit einem orangen Material.....	39
Abbildung 17: Implementierung der Materialspeicherung und Änderung	39
Abbildung 18: Nacher: ZoomCamera unabhängig vom anderen Komponenten im VR-Pawn Blueprint.....	40
Abbildung 19: Vorher: Zoom-Camera abhängig vom Motion Controller.....	40
Abbildung 20: Implementierung der Kamera-Bindung an den Motion Controller, der denn Zoom gerade aktiviert hat.....	41
Abbildung 21: Erste Version der Lupe mit Griff.....	41
Abbildung 22: Geöffnete Lupe mit Fadenkreuz und ohne Griff im VR-Modus	42
Abbildung 23: Material-Speicherung	42
Abbildung 24: Material-Markierung setzen.....	43
Abbildung 25: Material setzen	43
Abbildung 26: Implementierung des Zooms im VR-Pawn.....	44
Abbildung 27: Ändern des Event Server Posses Pawns	45
Abbildung 28: Android ist das Betriebssystem von der Oculus Quest. Deshalb wurde dies noch ergänzt.	46
Abbildung 29: Metadaten aus einem Excel-Dokument als CSV-Dateivorher	46
Abbildung 30: Über Texturen iterieren und passende zuweisen	46
Abbildung 31: Türrahmen Überschneidung	47
Abbildung 32: Türen verkleinern und positionieren	47

Pflichtenheft

Direkter PDF-Downloadlink über Redmine: [Pflichtenheft](#)

Lastenheft

Direkter PDF-Downloadlink über Redmine: [Lastenheft](#)

Produktbeschreibung

InfoStructure VR wurde im Rahmen des Bachelor-Software Projekts an der Westfälischen Hochschule Gelsenkirchen durch einen Zusammenschluss zwischen dem Fachbereich Umweltschutz und Gebäudetechnik (FB 1), sowie dem Fachbereich Informatik und Kommunikation (FB 3) aufgestellt. Thema dieses Projektes, ist es die Gebäudeinfrastruktur aus einem Gebäudemodell, welches mit Hilfe einer CAD-Software konstruiert wurde, visuell aufzubereiten und in einer virtuellen Umgebung begebar zu machen.

Intention zu diesem Projekt liefert Prof. Dr. Fieberg, der sich für die Bewerbung seines Master-Studienganges Building Information Modeling, eine visuell attraktive Möglichkeit gewünscht hat, Gebäudetechnik für außenstehende interessanter zu gestalten.

Das Finale Projekt umfasst einen definierten Arbeitsablauf, um ein von einer CAD-Software erstelltes Gebäudemodell samt enthaltener technischer Infrastruktur in die Unreal Engine 4 (UE4) von Epic Games zu überführen und anschließend eine nutzbare Anwendung zu generieren. Die Anwendung enthält eine Grundpalette an Steuerungselementen, sowohl für Desktop als auch für VR-Brillen, um das verwendete Gebäudemodell zu besichtigen und mit diesem zu interagieren.

In der Anwendung ist es möglich, durch Verteilung dieser an Dritte, mit multiplen Nutzern gemeinsam dasselbe Modell zu besichtigen.

Zu den Steuerungselementen zählen unter anderem, Funktionen zum Ausblenden oder Isolieren von Objekten für die Verbesserung der Übersicht, als auch die Möglichkeit sich aus der Gebäudetechnik wichtige interne Informationen ausgeben zu lassen. Die Navigation mit einem VR-Headset geschieht mittels Teleportation.

Im Produktumfang enthalten, ist:

- ein Unreal Engine 4 Projekt, als Grundgerüst für die Bedienung der Anwendung für Desktop und VR
- eine gesonderte Version des Projektes, welche explizit für die Oculus Quest entwickelt wurde.
- eine Software zur Extrahierung und Aufbereitung von Informationen der Gebäudeinfrastruktur
- eine Anleitung in Form eines Videohandbuches, zur Begleitung des Arbeitsablaufes.

Projektdurchführung

Methodik

Im Rahmen des Bachelor-Softwareprojekts wurde zu Beginn entschieden, dass klassisch vorgegangen werden soll. Demnach werden Prozessabläufe als eigenständige Abschnitte des Softwarelebenszyklus angesehen (vgl. Sommerville, S.56f). Das heißt, dass konkrete Meilensteine und Termine gesetzt werden und theoretisch wenig Spielraum für Änderungen bleibt.

Zunächst erschien uns das Wasserfallmodell als angemessenes Vorgehen, da dies häufig für kleinere Projekte verwendet wird. Beim Start war jedoch unklar, in welche Richtung sich das Unterfangen entwickelt. Das Team stand vor der Möglichkeit selbstständig ein Plugin für Unreal Engine zu implementieren oder auf das angekündigte Update von Unreal Engine mit dem Datasmith Plugin zu warten und sich anschließend mit der Aufbereitung der Informationen aus den IFC-Dateien, deren visueller Erscheinung und der Interaktion des Benutzers oder der Benutzerin in der virtuellen Realität zu beschäftigen. Es wurde beschlossen das Risiko einzugehen und auf das Update, welches voraussichtlich den Anforderungen für den IFC-Import entsprechen würde, zu warten

In der Zwischenzeit beschäftigte sich ein Teil der Gruppe mit dem Aufbau des IFC-Formats und der andere Teil mit den Möglichkeiten, die Unreal Engine allgemein und speziell für virtuelle Geräte, bot.

Nach der Erscheinung des neuen Datasmith Plugins von Unreal Engine änderte sich das Vorgehen zu einer Mischung aus inkrementeller Entwicklung und wiederverwendungsorientiertem Software Engineering. Das Team nutzte den von Unreal Engine zur Verfügung gestellten Collaborative Viewer als Basis für das Softwareprojekt. Auf diese Weise wurden von Version zu Version Funktionen übernommen, überarbeitet, entfernt oder neu entwickelt. Dies hatte den Vorteil, dass sich das Team nicht mit der Entwicklung von Standard Problemstellungen beschäftigen musste und die Zeit nutzen konnte benötigte Werkzeuge zu verbessern und auf die Bedürfnisse des Benutzers anzupassen.

Meetings wurden in der Regel einmal pro Woche abgehalten, um Aufgaben zuzuweisen, Schwierigkeiten zu besprechen und den aktuellen Stand mitzuteilen. Treffen mit Prof. Dr. Lux, Prof. Dr. Fieberg und Herrn Kollakowsky folgten keinem festen zeitlichen Rhythmus und fanden auf Anfrage statt.

Tasks, die als schwierig angesehen wurden, wurden meist in Zweiertteams bewältigt. Auf diese Weise konnte vorhandenes Wissen, beispielsweise bezüglich Unreal Engine, ausgetauscht und verschiedene Lösungsansätze gefunden werden.

Werkzeuge

Entwicklung

Unreal Engine 4

Wurde verwendet für die Realisierung des Projektes.

Vorteile

- Zur Verfügung gestellte Templates
- VR Support
- Datasmith als Plugin zum Import von IFC-Dateien
- Hohe Nutzung in der Industrie
- Gewaltiger Community Support
- Entwickler sind nah an der Community (Forum)
- Kostenlos

Nachteile

- Nicht anfängerfreundlich.
- Kein Gruppenmitglied hatte Erfahrungen mit dieser Spiel-Engine.
- Hohe Performance-Ansprüche
- Sehr komplex und umfangreich
- Kein Zugriff auf Teilversionierungen
- Empfindlich bei der Arbeit mit der Oculus Quest (Android)

Blueprints

Wurden verwendet, um objektorientierte Klassen oder Objekte in Unreal Engine zu definieren.

Vorteile

- Gute Übersicht
- Leicht nachvollziehbar
- Intuitiv bei der Suche von Funktionen

Nachteile

- Ungewohnte Oberfläche für herkömmliche Programmierer

SimpleBim 8

Wurde im Vorfeld für die Fehleridentifikation verwendet und anschließend für die Nachbearbeitung und den Export der IFC-Dateien genutzt.

Vorteile

- Prof. Dr. Fieberg arbeitet mit SimpleBim 8
- Mächtiges Tool, um IFC-Dateien zu validieren
- Eignet sich als CAD-Viewer für IFC
- 30-Tage Testversion

Nachteile

- Lizenzgebunden
- Veraltete Tutorials
- Schlechter Support

DDS-CAD Viewer

Wurde verwendet für die visuelle Darstellung der IFC-Dateien am Computer.

Vorteile

- Kostenlos
- Einfache Oberfläche

Nachteile

- Hohe Performance-Ansprüche

Netbeans 8.2

Verwendet für den Zugriff auf Metadaten und darauffolgende Ausgabe in aufbereiteter Form.

Vorteile

- Bereits vorhandene Erfahrung mit der Programmierumgebung
- Flexibel nutzbar (z.B. Java und HTML)
- Kostenlos

Nachteile

- Java JDK der Version 8 nur noch mit Oracle Account erhältlich

Blender

Wurde verwendet für die Trennung der Türbestandteile von dem CAD-Gebäude. Tür, Türrahmen, Türklinke und Scharniere wurden getrennt und als einzelne Meshes nach Unreal Engine 4 exportiert.

Vorteile

- Kostenlose Software
- Sehr viele Funktionen
- Arbeitet gut mit Unreal Engine 4 zusammen
- Viele Exportmöglichkeiten

Nachteile

- Nicht anfängerfreundlich
- Gewöhnungsbedürftige Bedienung

Microsoft Excel

Metadaten werden in Form einer Exceltabelle exportiert, um im Weiteren darauf zugreifen zu können

Vorteile

- Einfacher Zugriff aus programmiertechnischer Sicht
- Gute tabellarische Strukturierung der IFC-Daten
- Als Student ist die Lizenz für Microsoft Office kostenlos

Nachteile

- Keine

Design

Adobe Illustrator

Wurde verwendet für die Erstellung des Ingame-Menüs, des Tooltips und die Dokumentation der Steuerung.

Vorteile

- Viele Funktionen
- Export als Vektor-Datei möglich

Nachteile

- Aufgrund der teuren Lizenz ist die private Nutzung schwierig

Adobe Photoshop

Wurde verwendet für die Erstellung des Logos, des Startmenüs und für die Bearbeitung von Bildern.

Vorteile

- Viele Gestaltungsmöglichkeiten
- Ermöglicht genaue Arbeit
- Vertrautheit im Umgang

Nachteile

- Aufgrund der teuren Lizenz ist die private Nutzung schwierig
- Erstellung von Vektorgrafiken ist nicht möglich

Adobe InDesign

Wurde verwendet für die Erstellung des Plakats für Show&Tell.

Vorteile

- Viele Gestaltungsmöglichkeiten
- Ermöglicht genaue Arbeit

Nachteile

- Aufgrund der teuren Lizenz ist die private Nutzung schwierig

Organisation

Redmine

Redmine ist eine Projektmanagement-Software. Es wurde verwendet, um Tickets zu erstellen und zu dokumentieren, Zeiten zu buchen und Protokolle zu speichern.

Vorteile

- Kostenloser Service
- Übersicht über den aktuellen Stand des Projektes
- Leichte Identifikation der Mitglieder zu bestimmten Tasks

Nachteile

- Nicht anfängerfreundlich. Kein Gruppenmitglied hatte Erfahrungen mit einem Ticketsystem. Dies sorgte am Anfang für einige Missverständnisse und Fehleingaben.
- Begrenzte Rechte. Kommentare und Zeitbuchungen konnten bei Fehlern nicht nachträglich geändert werden. Dies schadet der Übersicht und sorgt für falsche Informationen.

TimeTree

Wurde verwendet als gemeinsamer Kalender für Terminplanungen und Zeitmanagement. Die Gruppenmitglieder trugen Ihre privaten Termine in den Kalender ein, welche die restlichen Mitglieder sehen konnten. So wurden freie Tage für Meetings bestimmt.

Vorteile

- Eigener Chat verfügbar
- Einfache Bedienung
- Übersichtlich
- Kostenlos
- Mit Android und iOS kompatibel

Nachteile

- Werbung bei kostenloser Version

Sciebo

Sciebo ist ein Cloud-Speicherdienst. Es wurden alle möglichen Dateien in einen gemeinsamen Ordner hochgeladen, der für alle Mitglieder zur Verfügung stand.

Vorteile

- Einfache Bedienung
- Verfügbar als iOS, Android und Desktop App
- Für Studenten 30 GB kostenloser Speicher, möglich auf bis zu 500 GB zu erweitern

Nachteile

- Schlechte Portierung auf iOS
- Autor von hochgeladenen Dateien nicht klar erkennbar

Kommunikation

WhatsApp

Wurde als Kommunikationsmittel verwendet.

Vorteile

- Schnelle und verschlüsselte Kommunikation
- Alle Mitglieder verfügten bereits über WhatsApp und hatten bereits Erfahrungen damit
- Es musste keine zusätzliche App oder Software runtergeladen werden

- Kostenlose App
- Mit Android und iOS kompatibel

Nachteile

- Keine klare Trennung zwischen Privatem und Arbeit

Slack

Slack ist ein Messaging-Dienst, der für die Kommunikation mit Herrn Kollakowsky und weiteren Kommilitonen, die im Labor tätig waren, verwendet wurde.

Vorteile

- Einfache und schnelle Kommunikation mit Herrn Kollakowsky
- Einfache Reservierung des Labors, da der Chat für alle sichtbar ist
- Kostenlose App
- Mit Android und iOS kompatibel

Nachteile

- Man bekam eine Nachricht im eigenen Gruppenchat, wenn in Redmine ein Ticket erstellt oder bearbeitet wurde. Dies führte teilweise zu Spam und so konnten Nachrichten, zum Beispiel von Herrn Kollakowsky, übersehen werden.

Dokumentation

Open Broadcaster Software Studio (OBS)

OBS ist eine Software für Aufnahme des Bildschirminhalts eines PCs. Es wurde verwendet für die Aufnahme der Anleitungs- und Dokumentationsvideos.

Vorteile

- Open-Source-Software
- Einfache Bedienung
- Sehr viele Funktionen und verschiedene Aufnahmemöglichkeiten

Nachteile

- Benötigt verhältnismäßig viel Ressourcen

Microsoft Word

Wurde verwendet für die Dokumentation des Projektes.

Vorteile

- Einfache Bedienung
- Als Student ist die Lizenz für Microsoft Office 365 kostenlos

Nachteile

- Begrenzte Gestaltungsmöglichkeiten

Vegas Pro 12

Vegas Pro 12 ist ein Videoschnittprogramm. Es wurde verwendet für die Bearbeitung der Anleitungsvideos.

Vorteile

- Sehr viele Funktionen

Nachteile

- Nicht anfängerfreundlich
- Einschränkungen beim Import der Formate
- Nicht kostenfrei

Produktbedienung

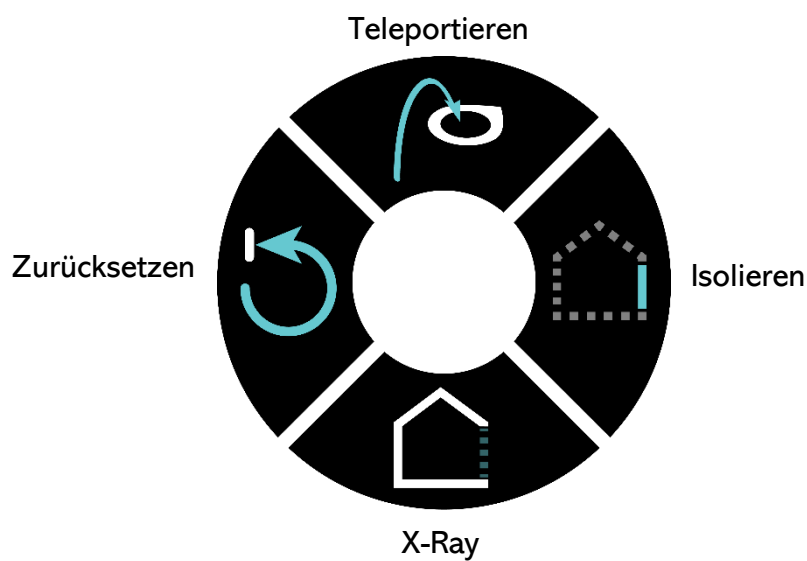
Videoanleitung:

Direkter Zip-Downloadlink über Redmine: [Anleitung](#)

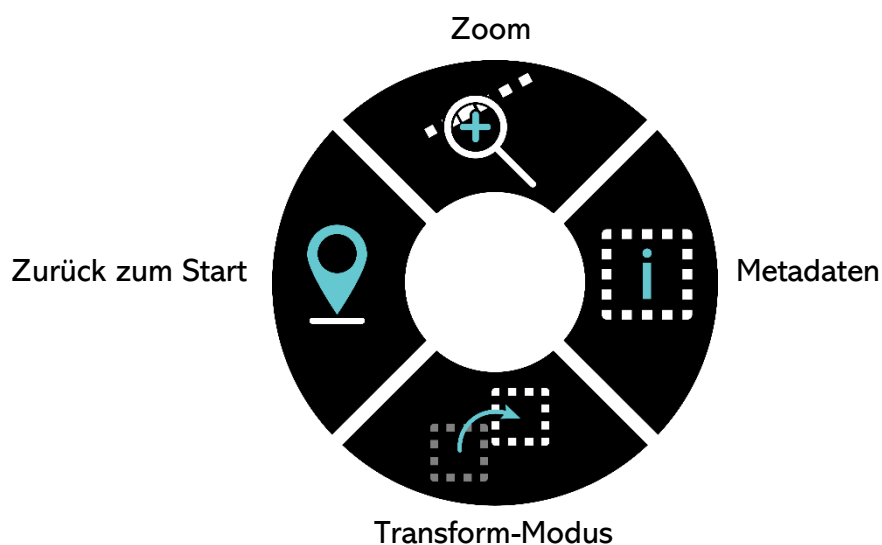
Steuerung

Tooltip

Das Tooltip ist nur im VR-Modus sichtbar

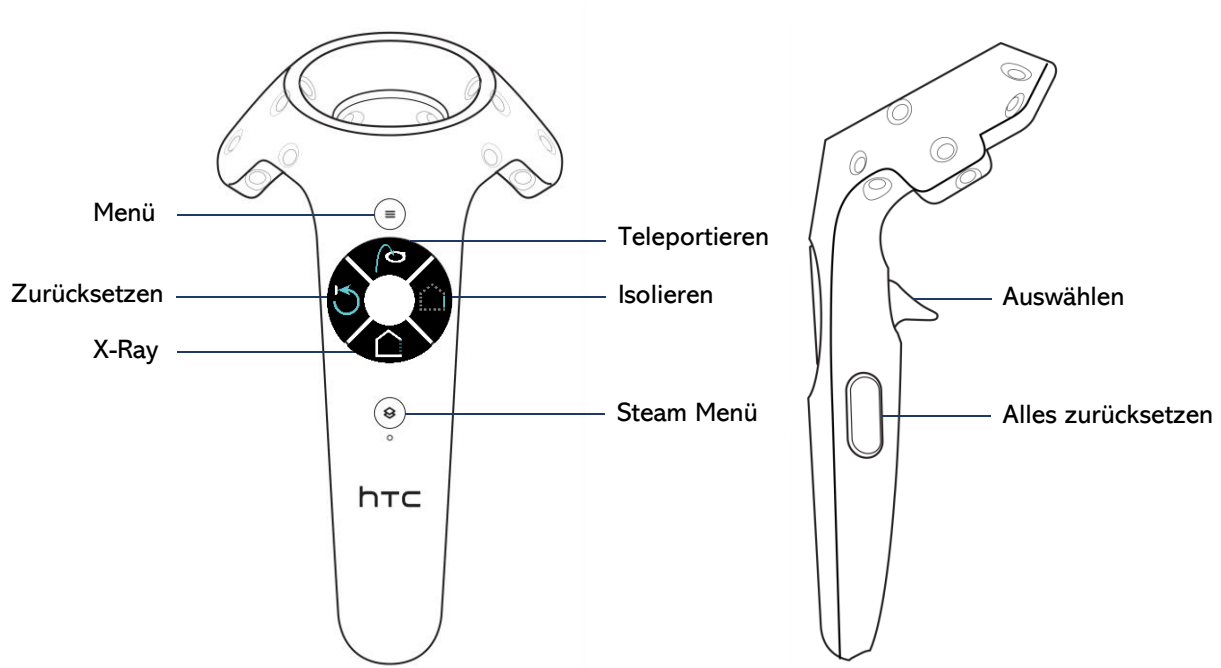


Menü



VR

HTC VIVE



Auswählen

Mit dieser Taste wählen Sie durch einen Laserpointer ein Objekt aus. Es erscheint eine farbige Umrandung um dieses. Nun können spezifische Aktionen durchgeführt werden. Wenn Sie dasselbe Objekt erneut auswählen, wird Ihre Auswahl aufgehoben.

Teleportieren

Das Teleportieren dient zur Fortbewegung in der virtuellen Anwendung. Wenn man die Teleportieren-Taste gedrückt hält, erscheint ein Kreis mit einem Pfeil, der in die Richtung deutet, an welche Sie springen werden, wenn Sie die Teleportieren-Taste loslassen. Beim Neigen des Motion Controllers können Sie die Blickrichtung, die Sie nach dem Sprung haben werden, festlegen.

Isolieren

Wenn Sie ein Objekt ausgewählt haben, können Sie mit der Isolieren-Taste alle anderen, nicht ausgewählten Objekte, durchsichtig erscheinen lassen.

X-Ray

Mit der X-Ray-Taste können Sie ein bereits ausgewähltes Objekt, transparent erscheinen lassen. Dadurch können Sie durch das Objekt durch gehen oder Objekte, die dahinter liegen, betrachten.

Zurücksetzen:

Nach Verwendung von „Auswählen“

Hebt die Auswahl auf. Es ist nichts mehr ausgewählt.

Nach Verwendung von „X-Ray“

Setzt alle zuletzt transparent gemachten Objekt wieder auf sichtbar.
Es setzt also das X-Ray vollständig zurück.

Nach Verwendung von „Isolieren“

Hebt Isolieren vollständig auf. Alle Objekte werden wieder sichtbar.

Während des „Transform-Modus“

Das ausgewählte Objekt springt an die ursprüngliche Stelle zurück.

Alles Zurücksetzen

Alle Ihre Aktionen werden vollständig zurückgesetzt.

Menü

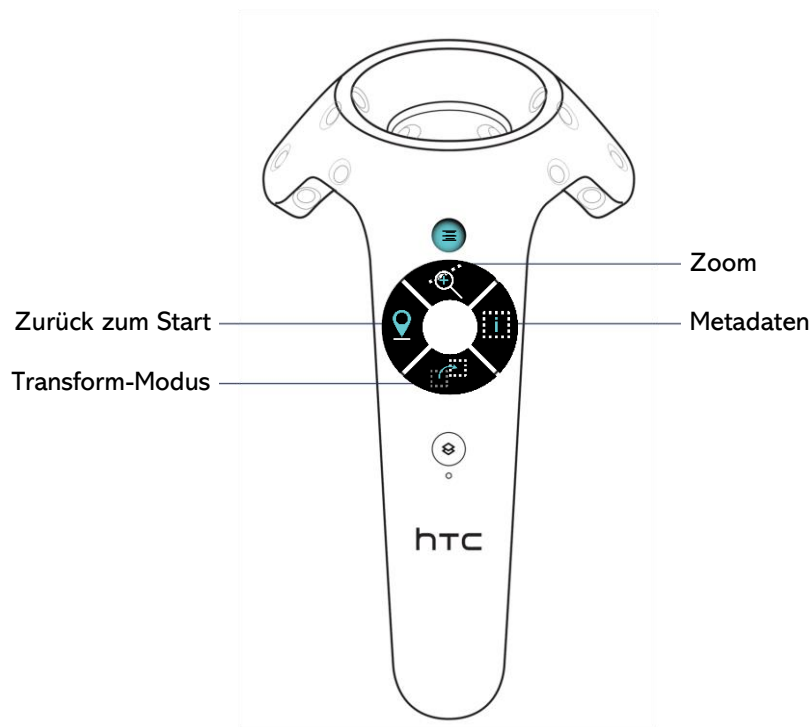
Öffnet das Menü am Controller. Der „Zoom“, die „Metadaten“, der „Transform-Modus“ und „Zurück zum Start“ stehen zur Verfügung. Bei erneutem Betätigen der Taste wird das Menü geschlossen.

Steam Menü

Mit dem Steam Menü können Sie die Anwendung oder das System direkt in VR schließen. Außerdem können Sie die Steam Einstellungen öffnen und den Desktop-Rechner mit den Controllern benutzen.

Steuerung im Menü

Es wird die Steuerung gezeigt nachdem die Menü-Taste gedrückt wurde.



Zoom

Es öffnet sich eine Art Lupe, die eine vergrößerte Ansicht der dahinter liegenden Objekte zeigt. Es ist ein Fadenkreuz zur besseren Auswahl gegeben. Wenn ein Objekt ausgewählt wird, erscheint es vollständig rot. Durch erneutes Betätigen der Menü-Taste wird der Zoom geschlossen.

Metadaten

Nachdem ein Objekt ausgewählt wurde, werden nach Betätigung der Metadaten-Taste, spezifische Informationen über das ausgewählte Objekt am Motion Controller angezeigt. Wenn ein anderes Objekt ausgewählt wird, werden die Informationen aktualisiert. Beim erneuten Betätigen der Menü-Taste werden die Metadaten nicht mehr angezeigt.

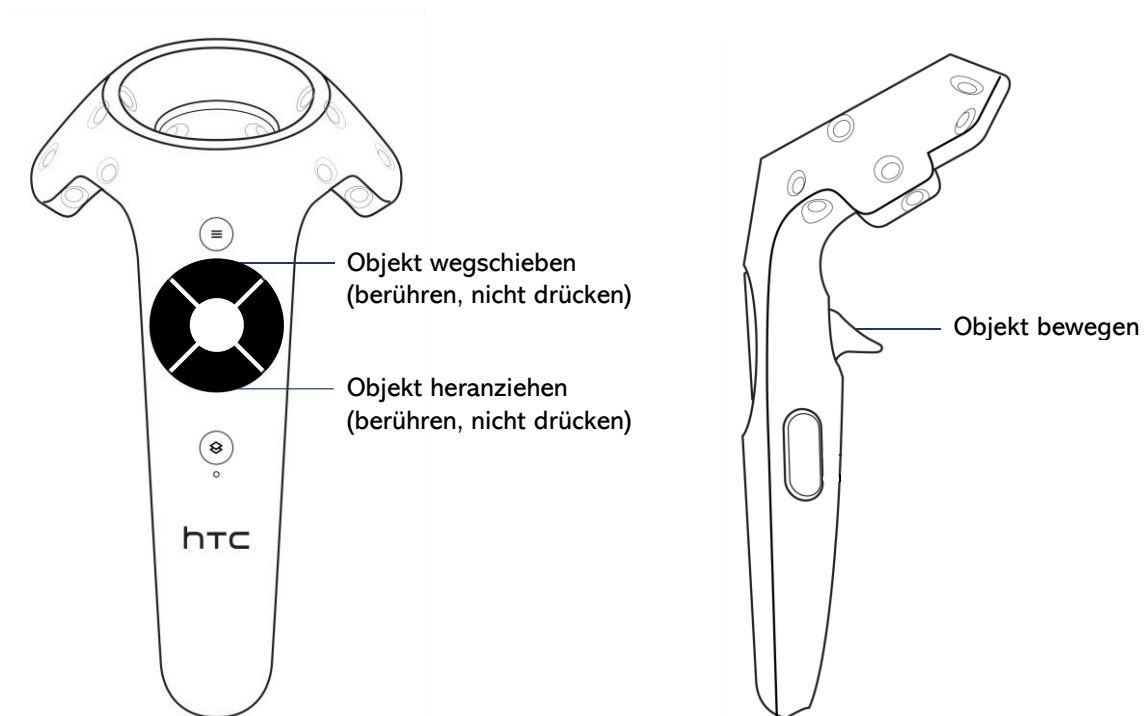
Zurück zum Start

Der Benutzer wird zurück an den Startpunkt geschickt. Aktionen werden nicht zurückgesetzt.

Transform-Modus

Im Transform-Modus können Objekte bewegt, herangezogen und weggedrückt werden. Der Laserpointer ist dauerhaft sichtbar. Der Transform-Modus wird durch erneutes Betätigen der Transform-Taste im Menü deaktiviert. Während des Transform-Modus ist „X-Ray“, „Isolieren“ und „Zoom“ nicht verfügbar.

Steuerung im Transform-Modus



Objekt bewegen

Mit gedrückter Auswahl-Taste können Objekte bewegt werden. Durch das Neigen des Controllers, wird auch das Objekt geneigt. Beim Loslassen der Taste bleibt das Objekt an der aktuellen Position.

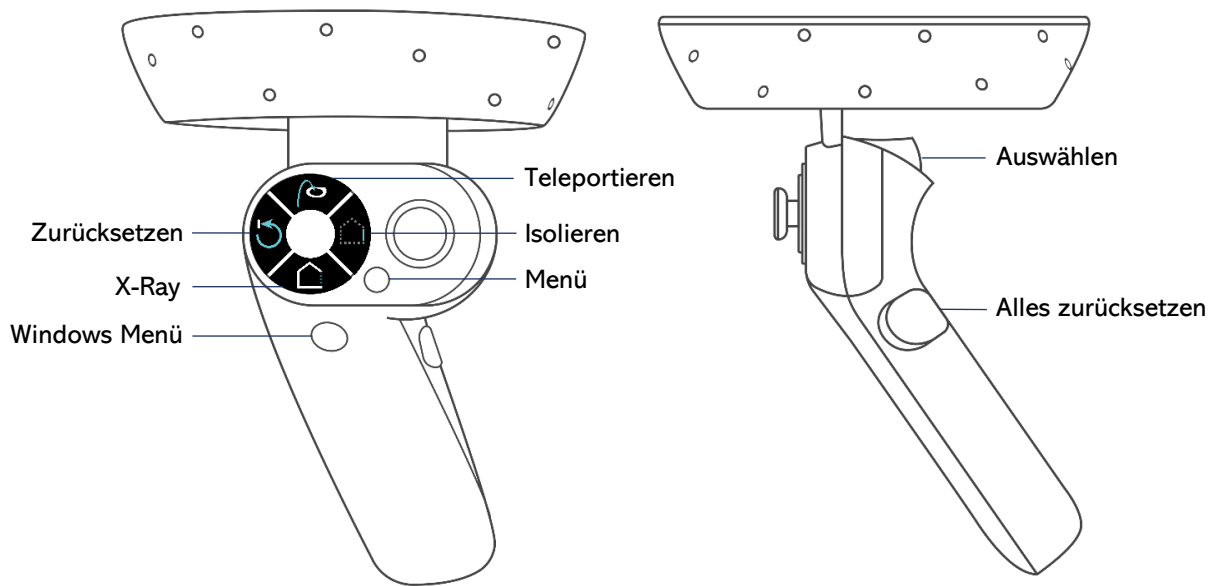
Objekt wegschieben

Durch das Berühren des oberen Touch-Kreises wird das Objekt weggeschoben. Beim Loslassen der Taste bleibt das Objekt an der aktuellen Position.

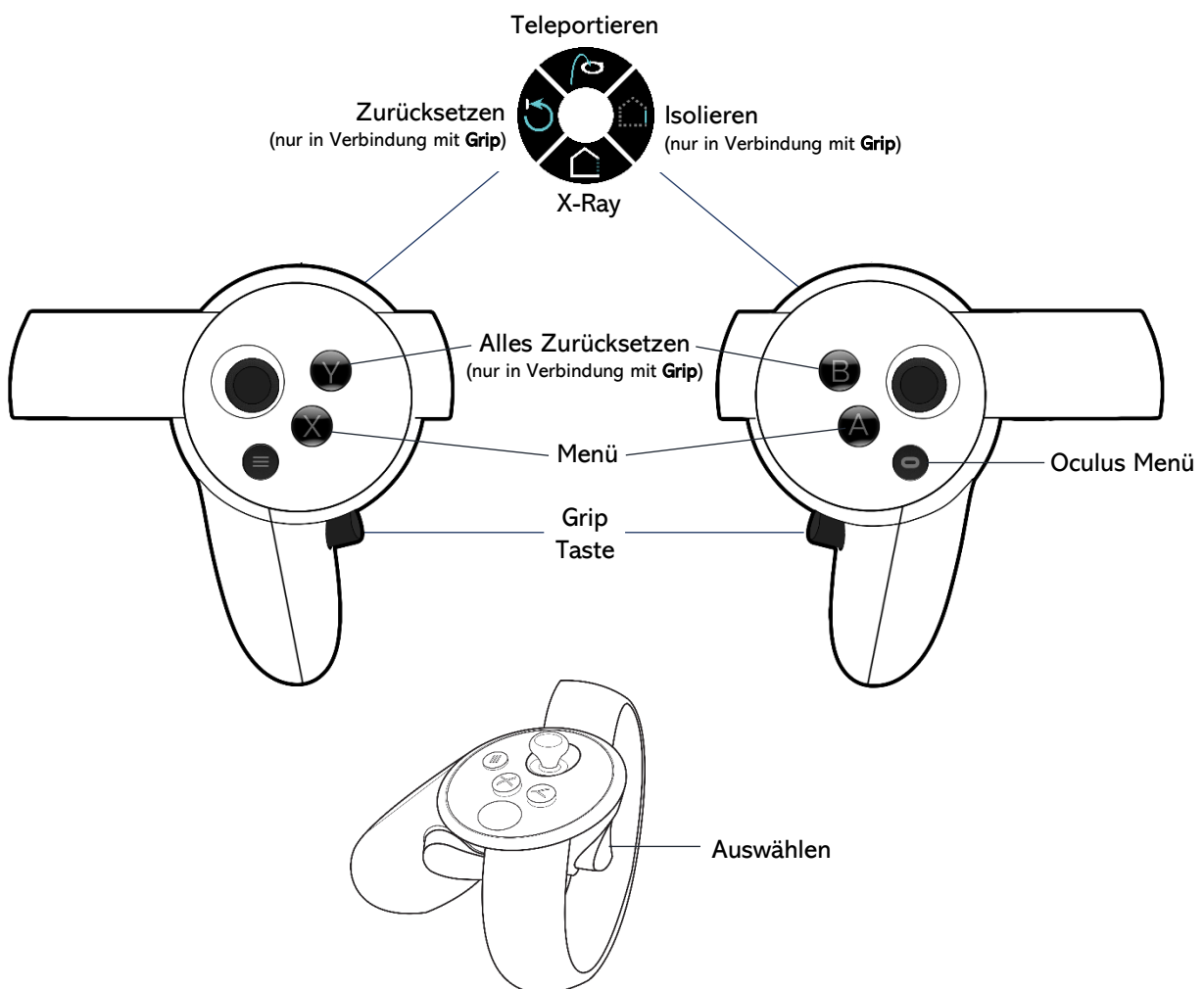
Objekt heranziehen

Durch das Berühren des unteren Touch-Kreises wird das Objekt herangezogen. Beim Loslassen der Taste bleibt das Objekt an der aktuellen Position.

Windows Mixed Reality Controller



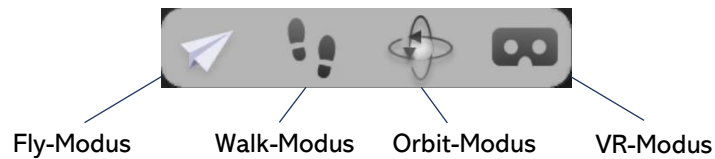
Oculus Quest Touch Controller



Um irrtümliche Befehle zu vermeiden, muss man für „Isolieren“, „Alles Zurücksetzen“ und „Zurücksetzen“ gleichzeitig die Grip-Taste betätigen. Dies gilt nur für die Oculus Quest.

Desktop

Benutzermodi



Benutzermodus wechseln

Erscheint oben rechts in der Anwendung. Der Benutzermodus kann durch das Anklicken auf das gewünschte Symbol mit der Linken Maustaste gewechselt werden.

Benutzermodus wechseln (Shortcuts)

Fly-Modus	-	1	-	Drücken
Walk-Modus	-	2	-	Drücken
Orbit-Modus	-	3	-	Drücken
VR-Modus	-	4	-	Drücken

Fortbewegung Walk-Modus

Vorwärtsbewegen	-	W	-	Drücken und Halten
Zurückbewegen	-	S	-	Drücken und Halten
Nach links bewegen	-	A	-	Drücken und Halten
Nach rechts bewegen	-	D	-	Drücken und Halten

Fortbewegung Fly-Modus

Vorwärtsbewegen	-	W	-	Drücken und Halten
Zurückbewegen	-	S	-	Drücken und Halten
Nach links bewegen	-	A	-	Drücken und Halten
Nach rechts bewegen	-	D	-	Drücken und Halten
Absteigen	-	X	-	Drücken und Halten
Aufsteigen	-	Y	-	Drücken und Halten

Fortbewegung Orbit-Modus

Rotationspunkt setzen	-	Mausrad	-	Drücken
Perspektive verschieben	-	Mausrad	-	Drücken und Halten
Heranzoomen	-	Mausrad nach vorne	-	Scrollen
Herauszoomen	-	Mausrad nach hinten	-	Scrollen

Allgemein

Auswählen	-	Linke Maustaste	-	Drücken
Umsehen	-	Rechte Maustaste	-	Drücken und Halten
X-Ray	-	E	-	Drücken
Isolieren	-	R	-	Drücken
Zurücksetzen	-	Q	-	Drücken
Alles zurücksetzen	-	Rücktaste	-	Drücken
Anwendung schließen	-	ESC	-	Drücken
Menü (an/aus)	-	Leertaste	-	Drücken

Shortcuts für Menü-Tools

Metadaten (an/aus)	-	F	-	Drücken
Transform-Modus (an/aus)	-	T	-	Drücken
Zurück zum Start	-	Enter	-	Drücken
Zoom (an/aus)	-	Z	-	Drücken
Heranzoomen	-	Pfeiltaste aufwärts	-	Drücken
Herauszoomen	-	Pfeiltaste abwärts	-	Drücken

Transform-Modus

Objekt bewegen	-	Linke Maustaste	-	Drücken und Halten
Objekt wegschieben	-	Mausrad nach vorne	-	Scrollen
Objekt heranziehen	-	Mausrad nach hinten	-	Scrollen

Projektaufbau

Klassen der Benutzerfigur

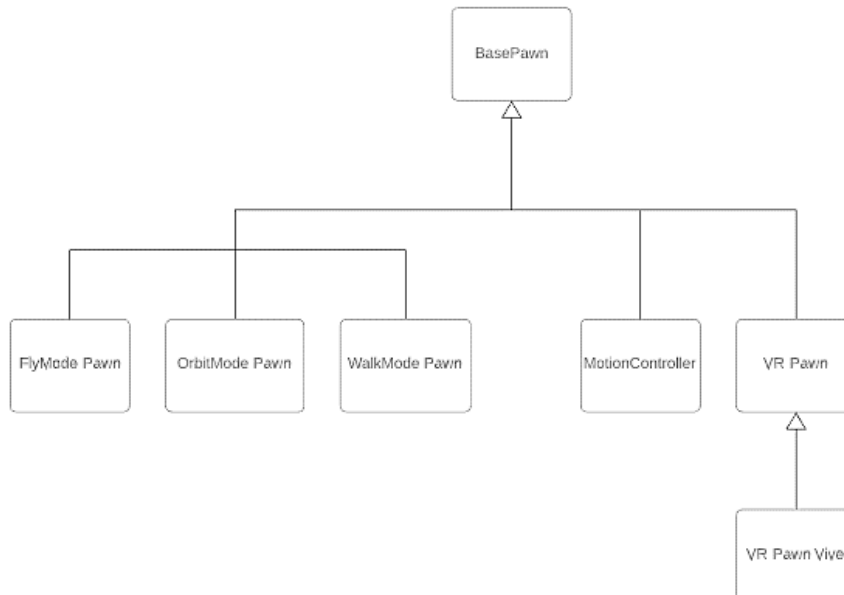


Abbildung 1: Klassen der Benutzerfigur im Projekt

Metadaten aus Excel (Java)

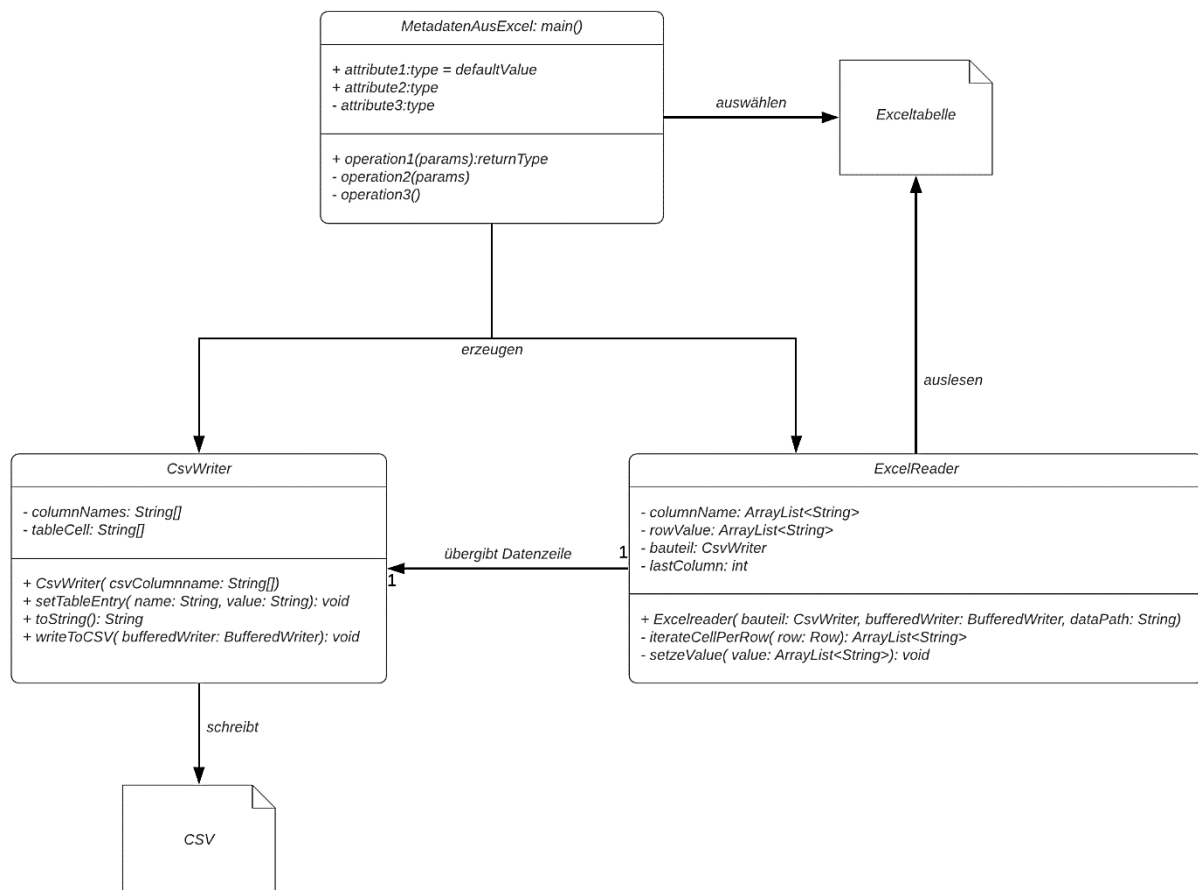


Abbildung 2: Metadaten aus Excel

Metadaten aus IFCXML (Java)

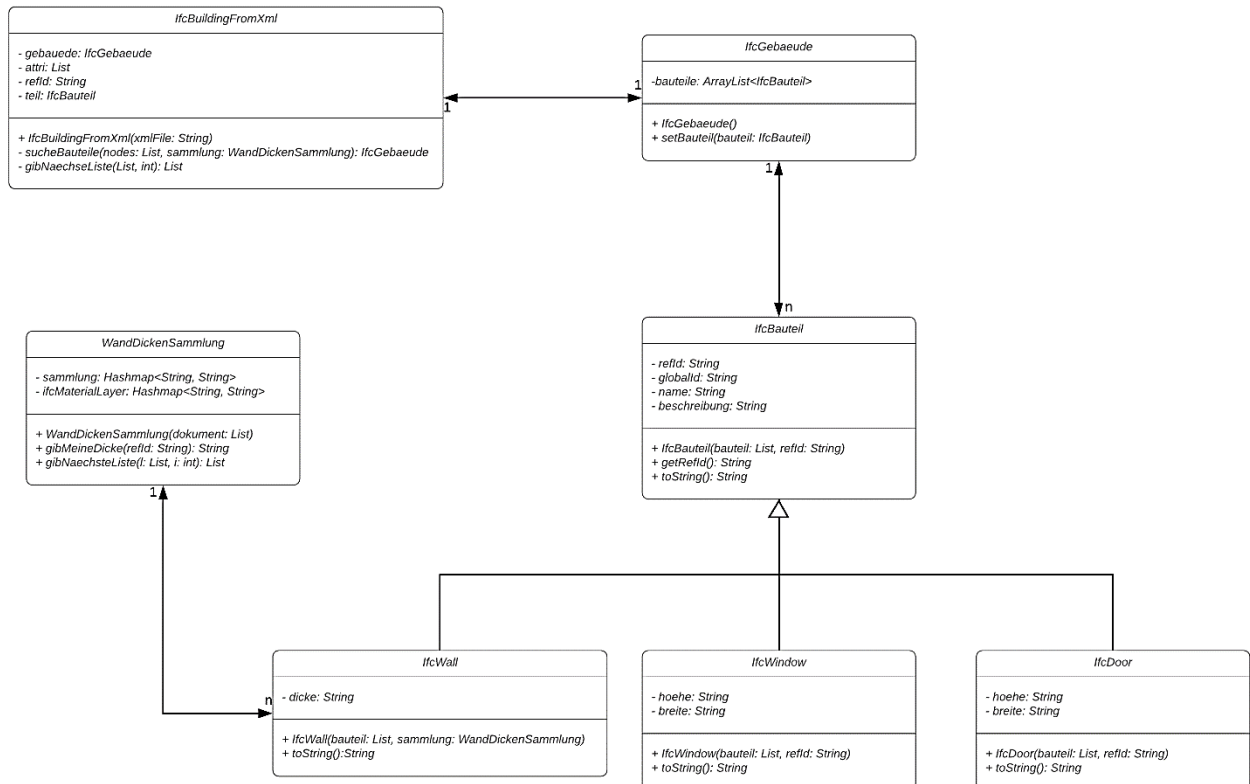


Abbildung 3: Metadaten aus IFCXML

User Interface Aufbau

Das User Interface beinhaltet sowohl das Menü als auch den Informationstext oder andere Darstellungsebenen, die als Rückmeldung für den Benutzer dienen.

Zustände des Menüs

Das Menü dient als Brücke zu den Funktionalitäten, die nicht häufig oder sofort zugänglich sein müssen, einen Modus einschalten oder die sowieso an den Motion Controller gebunden sind. Dabei interagiert es mit anderen User Interfaces, nämlich den Tooltips, Metadaten und dem Zoom. Diese Interaktion wird als Zustandsdiagramm in Abbildung 1 dargestellt.

Wenn es von einem in den anderen Zustand wechselt, beeinflusst es die Sichtbarkeit der anderen User Interfaces. Zum Beispiel werden die Tooltips unsichtbar, wenn man das Menü öffnet, da man die Funktionen, die auf dem Tooltip abgebildet sind, während des geöffneten Menüs nicht benutzen kann. Die Metadatenanzeige ist außerdem kein eigenständiges Objekt, sondern Teil des Menüs, weshalb innerhalb des Objektes Elemente in ihrer Sichtbarkeit angepasst werden müssen. Dies ist so konzipiert, da das Menü in VR in der Bewegung interpoliert wird und dementsprechend auch die Metadaten in der Bewegung.

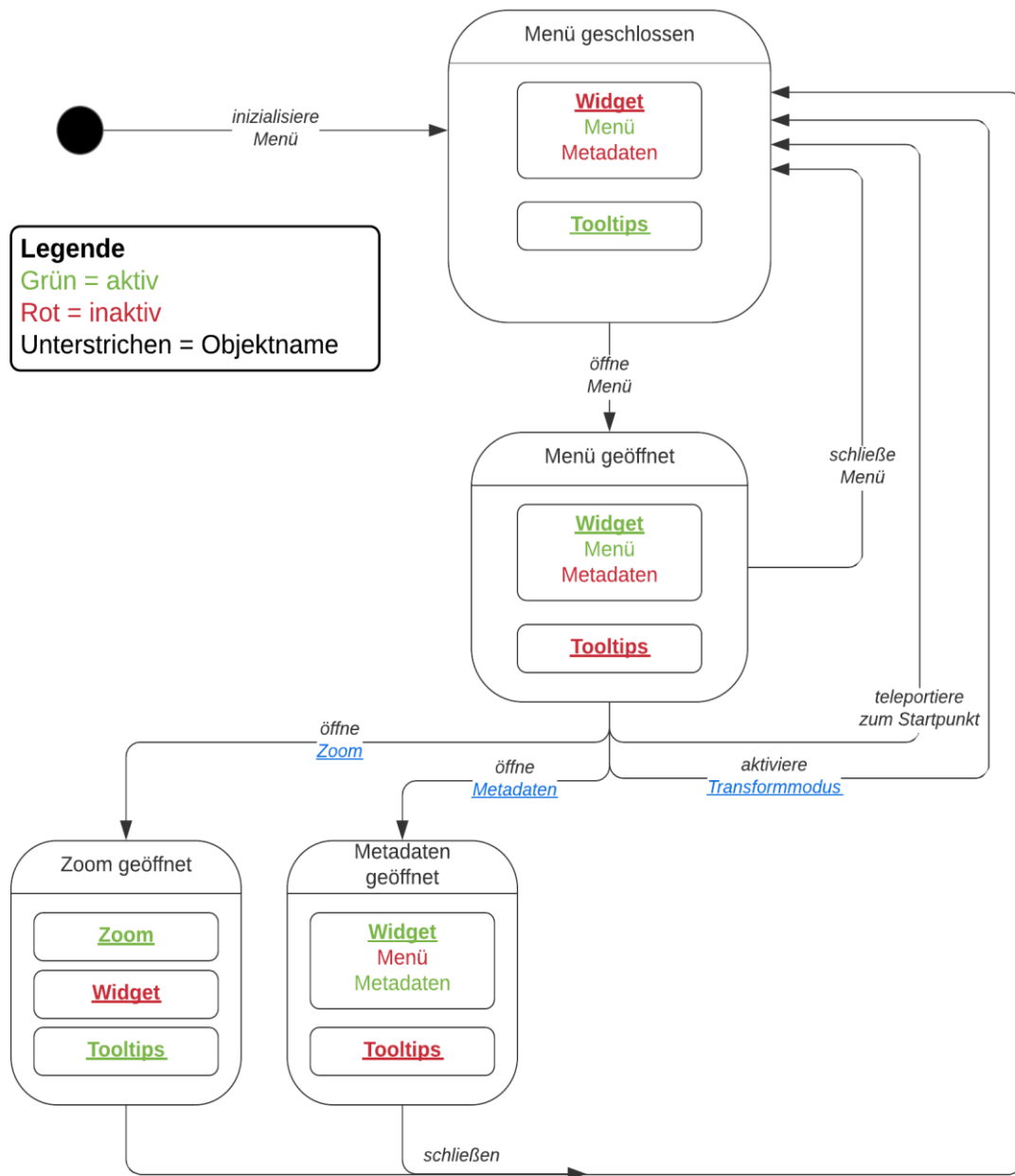


Abbildung 4: Zustandsdiagramm für das Menü

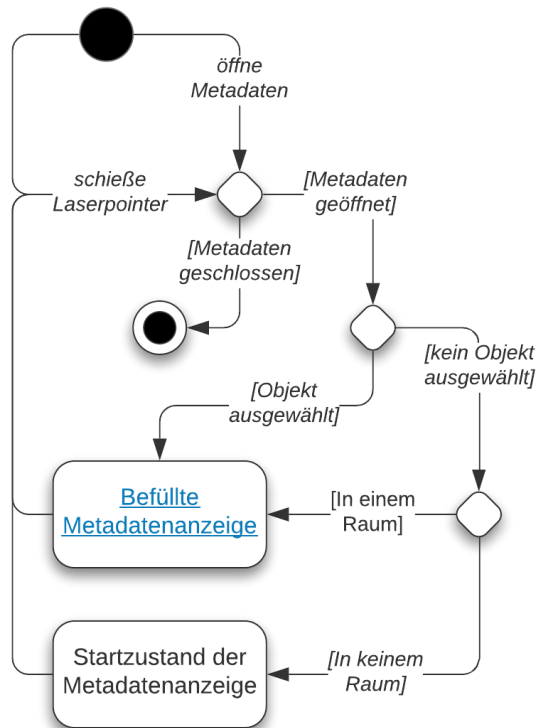


Abbildung 5: Zustandsdiagramm für die Metadatenanzeige

Zustände der Metadatenanzeige

Wenn man vom Menü aus die Metadatenanzeige öffnet, werden einige Bedingungen geprüft, um überflüssige Berechnungen vorzubeugen oder um festzustellen, ob die Anzeige befüllt werden kann.

Den Weg zu den zwei Zuständen der Metadatenanzeige, kann man in Abbildung 5 verfolgen. Zum Beispiel führt das Schießen des Laserpointers auch zu einer Aktualisierung der Metadatenanzeige, da man womöglich ein anderes Objekt ausgewählt oder das Aktuelle abgewählt hat. Dies wird so gehandhabt, um dem Benutzer eine Konsistente Darstellung der Metadaten zu bieten.

Außerdem kann auch ein Raum Metadaten enthalten. Diese werden angezeigt, wenn man sich in einem Raum befindet und kein Objekt angewählt hat.

Das Befüllen der Metadatenanzeige

Wenn alle Bedingungen erfüllt sind, um die Metadatenanzeige zu füllen, braucht man die Global-ID des Objekts, um sich die Metadaten aus der importierten CSV-Datei zu holen. Diese hat man vorher aus der IFC-Datei erstellt, um die Metadaten daraus zu extrahieren. In Abbildung 6 wird anhand eines Sequenzdiagramms gezeigt, wie sich die Metadatenanzeige aufbaut. Man schreibt sie für die entsprechende Zeile mit der Global-ID jeden verfügbaren Spalteneintrag als Zeile in den Datenblock. Welcher nach Beendigung der Schleife in die Metadatenanzeige geschrieben wird.

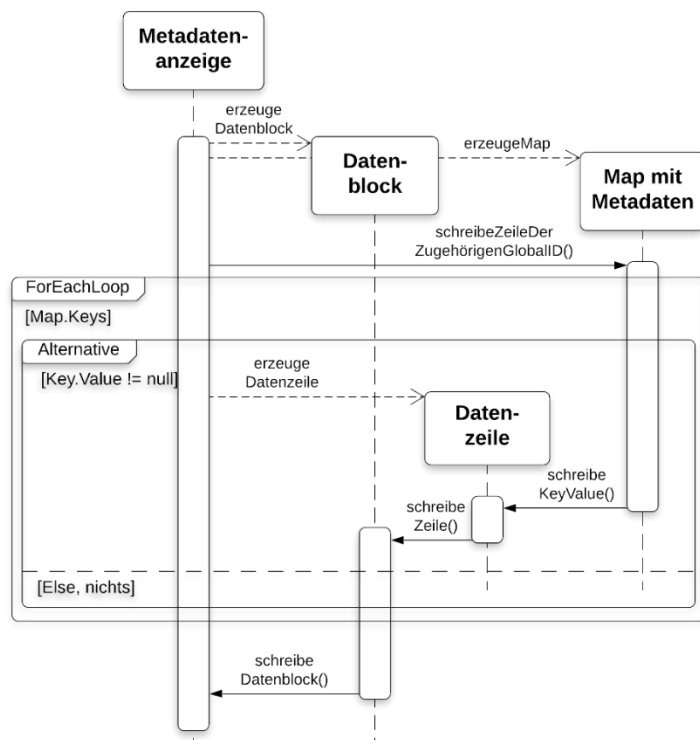


Abbildung 6: Sequenzdiagramm für die Metadatenanzeige

Abweichungen vom Pflichtenheft

Nicht erfüllte Pflichtenheft-Anforderungen

Im Folgenden finden Sie umrahmt Ausschnitte aus dem Pflichtenheft. Unterhalb wird der abweichende Istzustand näher erläutert und begründet.

Musskriterien

Die Anwendung

Metadaten sollen nach Möglichkeit an dem ausgewählten Objekt angezeigt werden.

Aus Konsistenzgründen werden Metadaten an den Motion-Controllern angezeigt. Zudem ist die Lesbarkeit immer gewährleistet.

Multi-User

Alle Handlungen der Benutzer sollen für alle sichtbar sein.

Alle Handlungen der Benutzer sollen für die Desktopmodi sichtbar sein. Der Multi-User zwischen mindestens zwei VR-Brillen funktioniert nur bedingt.

Wunschkriterien

Innenbeleuchtung durch Lichtschalter steuern.

Nicht vorhanden aufgrund von Performance Problemen der Beleuchtung und der aktuellen Aufbereitung der Metadaten. Es fehlen die Beziehungen unter den Objekten

Die Benutzung der Anwendung soll intuitiv gestaltet sein.

Personen, die noch keine VR Erfahrungen besitzen, könnten Einstiegsschwierigkeiten bei der Anwendung haben, aufgrund der Anzahl der Tools.

Die Objekte werden dynamisch mit erwartungsnahen Materialien angezeigt.

Die IFC-Datei enthielt inkonsistente Materialangaben. Größtenteils waren überhaupt keine vorhanden. Die Objekte werden dynamisch anhand ihrer IFC-Namen mit erwartungsnahen Materialien angezeigt.

Möblierung der ersten Etage des finalen IFC-Gebäudes.

Die Möbel würden die Bewegung im Gebäude behindern und bei der Auswahl der Objekte für die Metadaten stören.

Konfigurationen

/F060/ Ingame-Menü

Der Benutzer kann im Ingame-Menü die Funktionen X-Ray und Bookmark aktivieren.

Der Benutzer kann im Ingame-Menü die Funktionen Zoom, Metadaten, Transform-Modus, und Spawn-Bookmark aktivieren. Zur Erleichterung der Bedienung wurde die Zoom Funktion und der Transform-Modus integriert.

Tools

/F160/ Bookmark Funktion

Der Benutzer kann an vordefinierte Positionen springen.

Aufgrund der verschiedenen Gebäudemodelle sind keine eindeutigen Positionen ermittelbar. Deswegen wurde nur der Startpunkt als Spawn-Bookmark definiert.

Multi-User

/F310/ Kompatibilität der Geräte

Der Multi-User ist unabhängig von den verwendeten Geräten (Desktoprechner, HTC Vive, HTC Vive Pro oder Mixed Reality) möglich.

Der Multi-User ist nur zwischen maximal einer VR-Brille und einem oder mehr Desktoprechner möglich.

/F350/ Sichtbarkeit der Funktionen

Alle Benutzer sehen die getätigten Aktionen und deren Effekte untereinander

Ab zwei VR-Brillen sind die getätigten Aktionen und deren Effekte untereinander nicht sichtbar.

Ergänzungen zum Pflichtenheft

Transform-Modus

Mit dem Transform-Modus ist es möglich Objekte zu bewegen, wegzuschieben und an sich heranzuziehen.

Dieses Tool ermöglicht Objekte näher zu betrachten. Außerdem kann man Objekte an gewünschte Stellen verschieben um sich Veränderungen grob vorzustellen.

Designentscheidungen

Collab Viewer Template

Da keiner der Gruppenmitglieder Erfahrungen mit der Unreal Engine 4 hatte, informierten wir uns über bereits vorhandene Templates, die uns als Grundgerüst dienen sollten. Zur Wahl standen das VR Template und das Collab Viewer Template, welches uns Herr Kollakowsky zeigte. Der Collab Viewer warb mit dem Multi-User, inklusive der Einrichtung von Verbindungen und das Replizieren der Informationen zwischen mehreren Computern mit der VR oder am Desktop. Außerdem verfügte er über Funktionen, die sich besonders für das Gebäude eigneten. Der X-Ray beispielsweise erleichterte die Fortbewegung im Gebäude. Das VR Template hatte nur den „Teleport“ und das „Object Grabbing“, womit man Objekte nehmen und bewegen konnte. Da das VR Template kein Multi-User besitze, entschieden wir uns für den Collab Viewer.

Nach einigen Tests fanden wir beim Collab Viewer Fehler. Es gab Probleme mit dem X-Ray und beim „Zurücksetzen“ der bewegten Objekte im Transform-Modus. Im Laufe des Projekts fanden wir immer mehr Fehler im Collab Viewer, die uns viel Zeit zum Beheben kosteten.

Desktop-Modus

Wir haben uns entschieden den vorhandenen Desktop-Modus zu behalten, da dieser relativ gut funktionierte und sich für Benutzer eignete, die keine VR-Brille zu Verfügung hatten. Leider war der Desktop-Modus nicht perfekt. Er verfügte über einzelne Fehler und musste bei neuen Funktionen und Änderungen auch angepasst werden. Da aber der Multi-User zwischen zwei VR-Brillen nicht korrekt funktionierte, ergab sich der Desktop-Modus als eine gute Alternative. Denn der Multi-User zwischen einer VR-Brille und bis zu neun Desktop PCs verlief erfolgreich.

Menü

Altes Menü

Der Collab Viewer verfügte bereits über ein Menü. Dieses schien uns sehr kompliziert und unübersichtlich zu sein. Die Bedienung des Menüs erwies sich als sehr langsam. Außerdem wurden oft irrtümlich falsche Befehle ausgewählt. Das Menü war an dem Controller gebunden und bewegte oder drehte sich dementsprechend mit. Dies hatte den Nachteil, dass man den Controller ruhig halten musste, um das Menü lesen zu können.

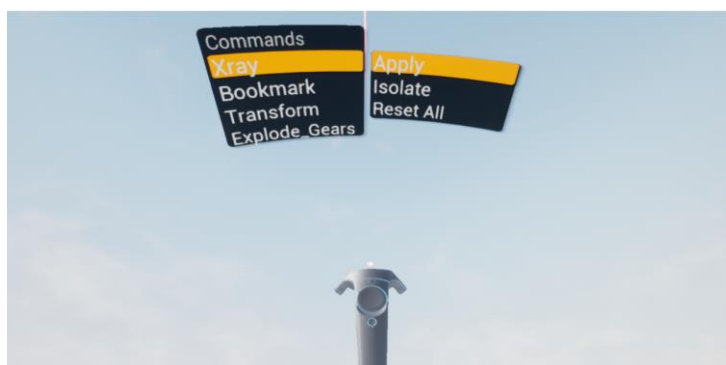


Abbildung 7: Controller mit dem alten Menü

Deshalb entschieden wir uns für ein neues eigenes Menü. Der „Zoom“ und die „Metadaten“ kamen hinzu, sodass wir insgesamt zehn Funktionen hatten. Beim Testen der Anwendung viel uns auf, dass bestimmte Funktionen deutlich öfter verwendet wurden als andere. Die Funktionen „X-Ray“ und „Zurücksetzen“ verwendeten wir für die Türen, um sich durch die Zimmer bewegen und die Sichtbarkeit der Türen zurückzusetzen zu können. Um diese Funktionen schneller aufrufen zu können, entschieden wir uns, diese außerhalb des Menüs zu platzieren. Dabei belegten wir alle vier Tasten des Touchpads. Dies hatte einen weiteren Vorteil, denn so wurde das Menü deutlich übersichtlicher und effizienter verwendbar.

Tooltip

Da die Funktionsbelegung nicht erkennbar war, erstellten wir ein Tooltip, das permanent am Touchpad des Controllers sichtbar war. Wir entschieden uns für eine Kreisförmige Oberfläche mit vier Tasten, die das Touchpad des HTC VIVE Controllers widerspiegeln soll. Die Funktionen wurden als Grafiken dargestellt. Das Tooltip besteht aus: „Teleportieren“, „Isolieren“, „X-Ray“ und „Zurücksetzen“.

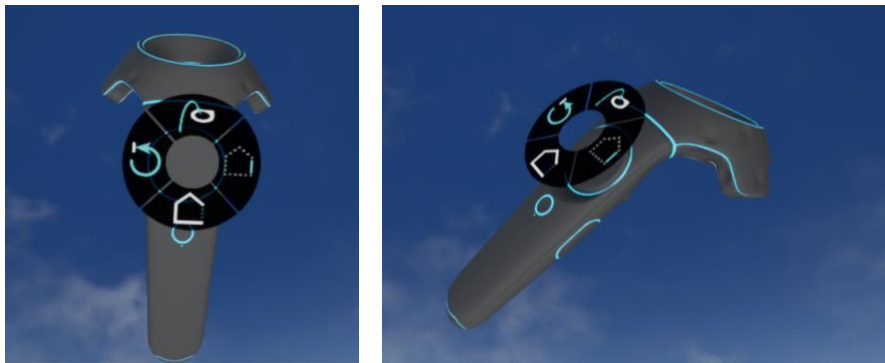


Abbildung 8: Controller mit dem Tooltip

Neues Menü

Das neue Menü hat dasselbe Design wie das Tooltip. Es besteht aus: „Zoom“, „Metadaten“, „Transform-Modus“ und „Zurück zum Start“. Der Aufruf dieser Funktionen ist deutlich schneller und einfacher als im alten Menü. Es muss lediglich die Menü-Taste betätigt werden und danach kann sofort die gewünschte Funktion mit dem Touchpad ausgeführt werden. Das neue Menü ist auch an dem Controller gebunden. Aber im Gegensatz zum alten Menü, verhält es sich trotz der Bewegung des Controllers sehr geschmeidig und gleichmäßig. Außerdem ist das Menü immer auf den Benutzer gerichtet, sodass es nicht mehr notwendig ist, die Controller ruhig zu halten.



Abbildung 9: Controller mit dem neuen Menü

Technische Entscheidungen

Wahl der Programmiersprache

Es war bekannt, dass sich C++ oder auch Python für die Beschaffung der Metadaten aus der IFC mehr eignen würde, da Unreal Engine selbst mit C++ Skripte arbeiten kann. Da keiner von uns Erfahrung mit den oben genannten Programmiersprachen hatte, hatten wir uns für Java entschieden, da es für die Bewältigung dieser Aufgabe vorerst ausreichte.

Wahl des Dateiformats für die Metadaten

Eine Vorläufige Version, um die Metadaten zu beschaffen, endete erfolgreich in der Ausgabe eines JSON Dokuments. Jedoch zeigte sich, dass das Einbinden in Unreal Engine 4, als Arbeitsschritt für den Endanwender, komplizierter war als vermutet. Im Hinblick darauf, die Arbeitsschritte möglichst einfach zu halten, entschieden wir uns die Metadaten in ein CSV-Dokument abzulegen. Der Vorteil wäre, dass sich die CSV durch einfaches Hineinziehen in das UE4-Projekt einbinden lässt, da wir die Tabellenstruktur bereits im Vorfeld definieren, die nötig ist damit UE4 diese richtig interpretiert.

VR-Brillen

Die HTC VIVE war die VR-Brille, mit der wir uns zu Beginn des Projektes am meisten auseinandergesetzt haben. Im weiteren Projektverlauf wurde unter anderem auch mit dem Windows-Mixed-Reality-Headset (WMR) an dem Projekt entwickelt. Dabei ist uns aufgefallen, dass die WMR, trotz der leichten Installation und Nutzung ohne zusätzliche Hardware, merkbare Probleme besitzt, den Abstand zum Boden klar erkennen zu können. Dabei können Symptome für Motion Sickness auftreten, die für Benutzer, die nicht in der Verwendung von VR-Anwendungen geübt sind, eine körperliche Beanspruchung bewirkt, die die Nutzung der Anwendung schwierig gestaltet.

Die stationäre Verwendung, und damit einhergehend die zuverlässige Erfassung der Bewegungen des Anwenders, ließ uns zu dem Schluss kommen, dass die HTC VIVE als primäres VR-Headset besser geeignet ist. Die Oculus Quest war ebenfalls ein Teil der Auswahl, da sie jedoch ein vom Computer unabhängiges Gerät ist (Standalone), wurde dafür ein gesondertes Projekt parallel mitentwickelt und ist daher gesondert aufzuweisen.

Beleuchtung

Im Rahmen der Visuellen Aufbereitung und dem daraus entstandenen Soll, das Modell möglichst realistisch darzustellen, haben wir versucht eine möglichst natürliche Beleuchtung zu verwenden. Da sich das Thema der Beleuchtung einen großen Platz in der Computergrafik verdient und nur sehr rudimentäre Kenntnisse darüber in der Projektgruppe vorhanden war, stießen wir schnell an die Grenze der vertretbaren Performanz der Anwendung. Mangels vorhandenen Wissensstands und aus Zeit-technischen Gründen haben wir uns dagegen entschieden eine Beleuchtung einzufügen.

Aufbau des Projekts

Wir haben uns dafür entschieden das Projekt auf Grundlage des Collab Viewer Templates aufzubauen, da es bereits einen großen Umfang an Funktionen darbot und ebenfalls die Schnittstelle enthalten war, um die Bedienung mit mehreren Nutzern im Netzwerk zu realisieren. Dadurch nahmen wir in Kauf, dass z.B. zusätzliche, nicht dringend notwendige Zusatzfunktionen, in die Projektpalette mit aufzunehmen, wie z.B. den Desktop Modus. Dies führte zwar zu Mehraufwand, jedoch rundete es den Gesamtumfang der Anwendung für den späteren Nutzen in unseren Augen ab.

Implementierung des Funktionsumfangs

Das Collab Viewer Template der Unreal Engine 4 stellte eine Palette an Funktionen für die VR Steuerung bereit, darunter die "Transform" Funktion, mit der man in der Lage war, Objekte in der grafischen Umgebung verschieben zu können. Wir befürchteten aber, dass diese Funktion schnell nur für Unordnung oder Verlust der Übersicht, während der Nutzung mit mehreren Anwendern, sorgte und haben uns kurzerhand entschieden diese Funktion auszulassen.

Im späteren Verlauf des Projektes, erschien uns jedoch diese Funktion in bestimmten Fällen als nützlich, wo gewisse Objekte, die außerhalb der Reichweite des Anwenders liegen, herangezogen werden können, um diese besser betrachten zu können. Deshalb wurde die "Transform"-Funktion in Form des "Transformmodus" wieder implementiert.

Einbindung eines Voice over IP-Dienstes (VoIP)

Es bestand die Idee eine Kommunikationsmöglichkeit zwischen mehreren Benutzern bereitzustellen. Diese sollte unabhängig von der Anwendung laufen, im Falle das es zu einem Absturz oder ähnlichem kommt. In der engeren Auswahl war die Software Mumble, eine frei erhältliche Software, mit der man über das Internet kommunizieren kann. Der Umstand, dass es noch nicht fehlerfrei möglich war, eine Sitzung, mit mehr als einen VR-Brillen-Nutzer zu führen, lies uns eine derzeitige Einführung von Mumble in den Projektumfang, als noch nicht sinnvoll erscheinen. Somit haben wir uns vorerst dagegen entschieden diese Möglichkeit einzubinden.

Technische Herausforderungen

Metadaten der IFC

Motivation

Da die Metadaten beim Import durch Datasmith nicht vollständig übermittelt wurden, mussten wir uns eine Möglichkeit überlegen, an die Metadaten gesondert zugreifen zu können. Die größte Schwierigkeit war zunächst, dass es keinen gängigen Ansatz gab, um diese Metadaten aus einer IFC-Datei direkt zu lesen. Es ist möglich eine IFC-Datei in einem herkömmlichen Texteditor zu öffnen. Der Inhalt steht dann im Klartext, aber die Syntax und die Vergabe von internen, nicht immer eindeutigen Nummern, ließ einen simplen Zugriff nicht ohne weiteres zu.

Wahl der Programmiersprache

Uns wurde nahegelegt, dass sich C++ mehr für diese Aufgabe eignen würde, da Unreal Engine 4 selbst damit arbeiten kann. Da sich leider keiner von uns mit C++ auskannte, entschieden wir uns für Java.

Nutzung von Drittanbieter Software

Um aus der IFC-Datei eine Datei zu erhalten, die einem XML Standard folgt, damit anschließend darüber iteriert werden kann, griffen wir zunächst auf die Software XBimExplorer zurück. Der Vorteil war es, dass diese Software nicht nur die IFC-Datei lesen und darstellen, sondern diese auch als IFCXML-Datei exportieren konnte.

Ein großes Problem zeigte sich jedoch im Projektverlauf, denn andere IFC-Dateien, die zu einem Späteren Zeitpunkt verwendet wurden, führten zum Absturz der Software, sobald die IFC-Datei gelesen wurde. Somit war dieser Weg nicht mehr zuverlässig und wir mussten eine andere Lösung finden.

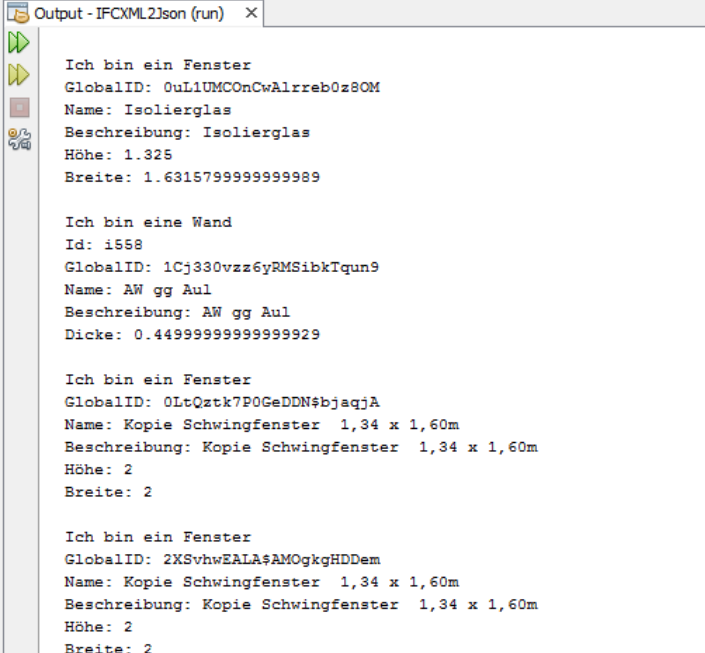
Während der Fehlersuche erhielten wir von Prof. Dr. Fieberg, auf unsere Anfrage hin, die einzelnen Gebäudetechnikpläne separat als IFC-Dateien.

Prof. Dr. Fieberg wies darauf hin, dass die einzelnen Modelle unterschiedliche Koordinaten des Ursprungs aufweisen und diese händisch nachgebessert werden müssten. Dabei empfiehlt er die Software „SimpleBim“ zu verwenden. SimpleBim ist ein Tool, um IFC-Dateien nach eigenen Anforderungen zu validieren und zu bearbeiten. Eine Herausforderung war hierbei, dass uns nur eine 30 Tage Testversion zur Verfügung stand, die Benutzerkonten gebunden ist, d.h. dass zur Aktivierung einer Testlizenz, immer ein neuer Benutzer diesen am gleichen Rechner aktivieren kann. Dies stellte sich als kleine Unannehmlichkeit in der Einarbeitung heraus.

Durch die Einarbeitung in SimpleBim, erkannten wir, dass die Software selbst in der Lage war, IFC-Dateien zusammenzufügen als auch auseinander zu ziehen und als neue IFC-Dateien abzuspeichern. Das Interessantere war jedoch, dass es auch möglich war, die IFC-Datei als Excel Tabelle zu exportieren. Die Daten wurden dadurch bereits in eine Struktur überführt, in der es insgesamt leichter war, die Daten zu finden und zu filtern, die wir benötigen. Da Prof. Dr. Fieberg ohnehin mit SimpleBim arbeitet und somit keine extra Software nötig war, wurde SimpleBim Teil des vorläufigen Workflows.

1. Erarbeitung des Algorithmus für IFCXML

Im Späteren Verlauf war es möglich mit Java über die IFC-Datei, die vorher als IFCXML exportiert wurde, zu lesen. Ein Problem war, dem grundsätzlichen Aufbau einer IFC-Datei geschuldet. Denn nicht jedes Objekt hat alle Informationen in ihrer Struktur eingespeichert. Während fast alle Bauteile eines Gebäudes die Informationen über ihre eindeutige Globale ID besitzen und in der Regel auch ihren Namen und Beschreibung, gibt es Unterschiede, was spezifizierende Daten angeht. Zum Beispiel steht bei jeder Tür und jedem Fenster die Information über ihre Maße. Bei einer Wand jedoch nicht wie dick diese ist oder welchen Wärmedurchlass sie besitzt. Diese Informationen sind in Objekten ausgelagert, die eine Reihe von Bauteilen über eine interne vergebene ID referenziert, die über genau diese Informationen verfügen sollen (z.B. die gleiche Wanddicke).



```

Output - IFCXML2Json (run) x
Ich bin ein Fenster
GlobalID: 0uL1UMCOncwAlrreb0z8OM
Name: Isolierglas
Beschreibung: Isolierglas
Höhe: 1.325
Breite: 1.6315799999999999

Ich bin eine Wand
Id: i558
GlobalID: 1Cj330vzz6yRMSibkTqun9
Name: AW gg Aul
Beschreibung: AW gg Aul
Dicke: 0.449999999999999929

Ich bin ein Fenster
GlobalID: 0LtQztk7P0GeDDN$bjaqjA
Name: Kopie Schwingfenster 1,34 x 1,60m
Beschreibung: Kopie Schwingfenster 1,34 x 1,60m
Höhe: 2
Breite: 2

Ich bin ein Fenster
GlobalID: 2XSVhwEALA$AMogkgHDDem
Name: Kopie Schwingfenster 1,34 x 1,60m
Beschreibung: Kopie Schwingfenster 1,34 x 1,60m
Höhe: 2
Breite: 2
  
```

Abbildung 10: Extrahierte Metadaten aus einem IFCXML-Dokument

Somit hat sich zunächst eine sehr grobe Arbeitsweise des Algorithmus ergeben, denn um alle Daten richtig den Objekten zuweisen zu können, musste man händisch die IFCXML-Datei analysieren und sich für jede Objekt Klasse die Struktur klar machen, wie die Informationen hinterlegt werden. Welches bis zu einem gewissen Stadium des Projektes nur für Fenster, Türen und Wände durchgeführt wurde. (siehe Abbildung 10 oben)

2. Erarbeitung des Algorithmus für Exceltabelle

Der Algorithmus war sehr einfach zu Implementieren. Jedoch kam das Problem auf, dass Kommata in den Datensätzen als Trennzeichen interpretiert wurden, wenn die anschließende CSV-Datei in die Unreal Engine importiert wurde. Da in der ersten Version des Algorithmus die FileWriter-Klasse verwendet wurde, um die Datensätze zu schreiben, gab es keine Möglichkeit die Formatierung anzupassen. Somit musste der Algorithmus nochmal überarbeitet werden und statt dem FileWriter, der BufferedWriter verwendet werden. Weiterhin kann es passieren, dass es doppelte Einträge gibt da die Exceltabelle mehrere Seiten besitzt, die für jede Klasse angelegt werden und dadurch auch Objekte mehrmals aufkommen können. Dies hat sich aber nicht als schwerwiegendes Problem gezeigt, da Unreal Engine 4 diese bereits zusammenfügt.

Weiterhin musste der Algorithmus so erweitert werden, dass man als Anwender selbst die Datei aussuchen und ggf. auch mehrere Dateien zu einer CSV-Datei zusammenfassen kann.

Anmerkung

Nach der Umstellung, die Metadaten aus der von *SimpleBim* erstellten Exceltabelle zu einer CSV-Datei zu konvertieren, ergab ein Gespräch mit Prof. Dr. Fieberg, dass die CAD-Software, die er verwendet, überraschenderweise in der Lage war das Modell in XML zu exportieren. Da es später erkannt wurde und der XML-Standard sich vom IFCXML-Standard unterscheidet, haben wir uns aus Zeitgründen dagegen entschieden den Algorithmus neu zu entwickeln.

Datenstruktur der Extrahierten Metadaten

Um die extrahierten Metadaten in die Unreal Engine einzupflegen, wurde zunächst überlegt diese Daten in ein JSON Dokument abzuspeichern. Vorteil wäre die ähnliche Semantik zur Objektorientierten Programmstruktur des ersten Algorithmus (s.o.).

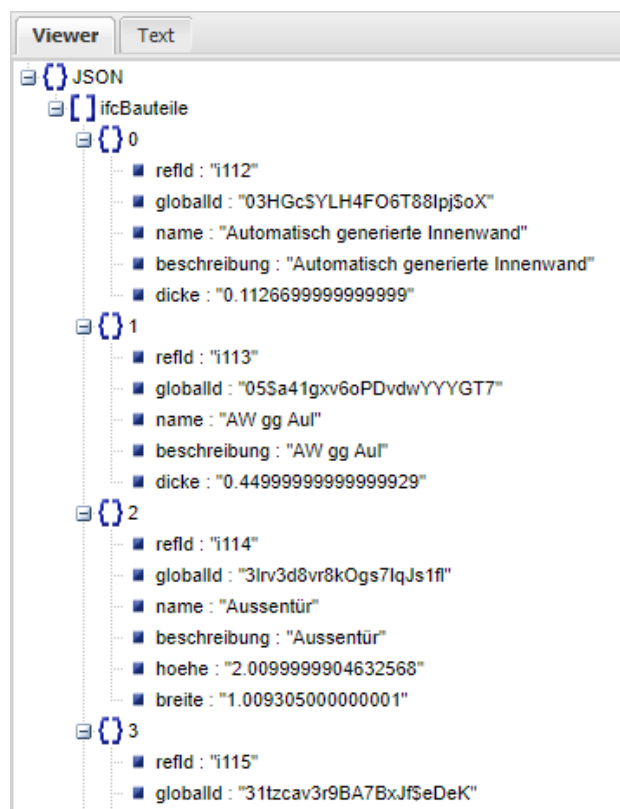


Abbildung 11: Ausgegebene JSON-Datei in einem Viewer

Im Nachhinein hat man sich dafür entschieden, die Daten als CSV-Datei abzuspeichern, da man diese direkt in das Projekt ziehen konnte und somit den Arbeitsaufwand des Nutzers minimiert. Ein Nachteil entstand dabei durch den Inhalt der Metadaten selbst. Denn oft wurden Zahlenwerte in Namen benutzt, die Kommazahlen verwendeten. Diese werden jedoch auch zur Spaltentrennung in einem CSV-Dokument verwendet, was zu Fehlinterpretationen der Tabellenstruktur führte.

```

184 1oeQccdi97vxhoRdBvb89A, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
185 32Mx4Im5j0Iud$GK3XAfgo, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
186 3d3MiLa09DRBwfVJRWgEOD, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
187 3kupGvt0v3rfwl_wHzwAWN, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
188 2o1OgloRl0Wf7LxmbhHdTc, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
189 0J9umnH4jBAOBMDrz7qxKU, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
190 3CN8GDjBL4Cfbai_MqE5aZ, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
191 0h502B76L3NQh9CNseP0uV, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
192 03AAGqOHDDUf1NB6YEKFuc, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
193 3xLYDyY7vAbP8oxHxRqZdL, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
194 0qDT3d$1zBYQ_plcQmg84F, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
195 2ejQ9bLuf8HBzf0li2UJko, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
196 2ZTIyD_WfDbO7z7R9DFALX, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
197 0dRYuW5iv49RkIx9C$KpQf, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
198 3k0Lu_hIn6VBCeWA51TzNO, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
199 1KHLQkAN5BzB6i8F$u6bmd, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
200 1KpcyE8$15FQbdfdl1lTOiwV, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
201 1EjZy03cnFuQAsKvRmqvHl, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
202 2XUU4ReTLEcRr5sp_zyj6n, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
203 1qeRaCOilA3R$zRvqZ$shDu, Dach,,, 0.1,,,,,,,,,,,,,Raumdecke,,,,,,,,,
204 0cDudrYeL7UPGB$Oogh7D7, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
205 22VN1ZLyLB4P3Th2c7jkjD, Dach,,, 0.1,,,,,,,,,,,,,Raumdecke,,,,,,,,,
206 0XhmND2OP9meBXQgvXqjKF, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,
207 09KPF$rxDFCP0lTUGtqgkH, Dach,,, 0.1,,,,,,,,,,,,,Raumdecke,,,,,,,,,
208 0XuJtpdwT5wg7JAj5j$HvU, Fußbodenheizung Erdreich 1.25m²K/W (min DIN EN 1264-4) ,,, 0.2,

```

Abbildung 12: Metadaten aus einem Excel-Dokument als CSV-Datei

Um eine CSV-Datei in die Unreal Engine zu laden, muss man im Vorfeld die Tabellenstruktur per Hand definieren. Dies kann weitere Ergänzungen der Datensätze erschweren.

Zoom in der VR

1. Problem: Vergrößerung und Einstellung der Sichtbarkeit

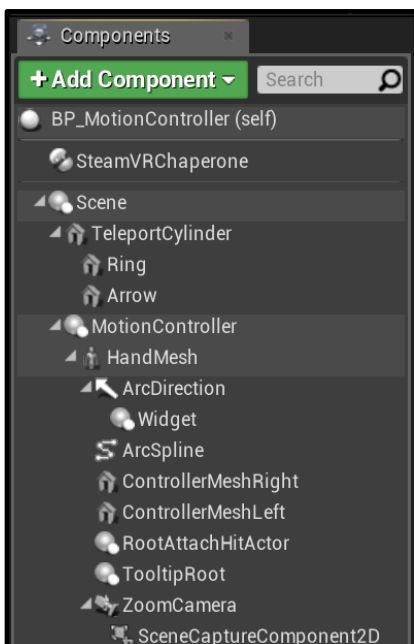


Abbildung 13: Kamera im Motion Controller Blueprint abhängig vom Motion Controller

Die Vergrößerung des Sichtfelds scheiterte zunächst, da das Sichtfeld nicht mit dem gleichen Blueprints wie im Desktop-Modus vergrößert werden konnte. Daraufhin kam die Idee auf, im VR-Modus eine Lupe zur Vergrößerung zu verwenden.

Hierfür wurde eine Kamera Komponente an den Motion Controller per View Port hinzugefügt. Ein SceneCaptureComponent2D wurde mit der Kamera verknüpft (s. Abb. 13). Aus dem RenderTarget wurde ein Material erstellt, welches auf ein Widget übertragen wurde.

Ursprünglich wurde ein Widget Component im Motion Controller Blueprint hinzugefügt (s. Abb. 13) und die Ansicht vergrößert. Hierbei gab es Schwierigkeiten mit dem An- und Ausschalten der Sichtbarkeit des Widgets. Die Lösung bestand darin, nicht mit einem Widget Component, sondern mit Widget Blueprint zu arbeiten und anschließend mittels Child Actor im Motion Controller auf das Zoom Widget zuzugreifen. Auf diese Weise konnte die Sichtbarkeit in der Desktop Anwendung erfolgreich gesteuert werden.

Beim Testen der Anwendung in VR sind Probleme aufgetreten. Die Sichtbarkeit konnte nicht ein- und ausgeschaltet werden und das Widget war am Boden befestigt (nicht wie ursprünglich am Motion Controller). Die Probleme wurden erfolgreich gelöst, indem der Widget Actor im VR Pawn Blueprint implementiert wurde, anstelle vom Motion Controller BP. Jedoch hat die Qualität der Widget Anzeige erheblich abgenommen (s.Abb.13).



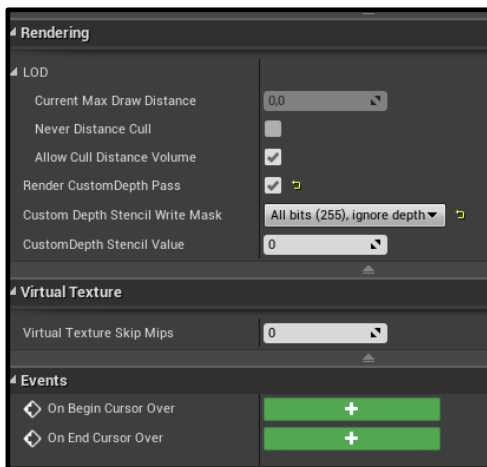
Abbildung 14: Anzeige des Zoom Widgets mit schlechter Qualität

2. Problem: Qualität

Überarbeitung der Zoom Implementierung:

- Plane Component als Medium für den Zoom im Actor, anstelle von einem Widget
- Unter RenderTarget Textur „Shading Model“
 - ➔ „Unlit Material“ nicht mehr „User Interface“
- Platzierung des Zoom Actors in den VR-Pawn Blueprint, nicht mehr im Motion Controller BP
- Bindung des Zooms an die Motion Controller erfolgt im VR-Pawn
- Verwendung eines Plane Components, so sind keine Umwege über ein Widget nötig.
 - ➔ Bessere Qualität und die Sichtbarkeit der Anzeige kann an- und wieder ausgeschaltet werden

3. Problem: Custom depth als Markierung



Durch die Zoom Fläche scheint die Umrandung der Markierung durch. Custom depth, welches für die Umrandung zuständig ist, wird von der Kamera bzw. dem Scene Component nicht angezeigt.

Das Durchscheinen wurde mit „Custom Depth Stencil Write Mask“-> ignore Custom depth unter den Rendering Einstellungen behoben (s. Abb. 15). Anschließend wurde nach Möglichkeiten, die Markierung durch Custom depth anzeigen zu lassen, gesucht und verschiedene Einstellungen des Post Process Materials und deren Auswirkungen auf dem Zoom getestet.

Abbildung 15: Lösung: Custom Depth Stencil White Mask: All bits (255), ignore depth

Als Workaround diente die Materialänderung als Markierung, da Custom Depth nicht im Scene

Component angezeigt wird. Das Material des Meshes wird zu einem Orangenem geändert, wenn das Objekt mit dem Zoom ausgewählt wurde (s. Abb. 16).

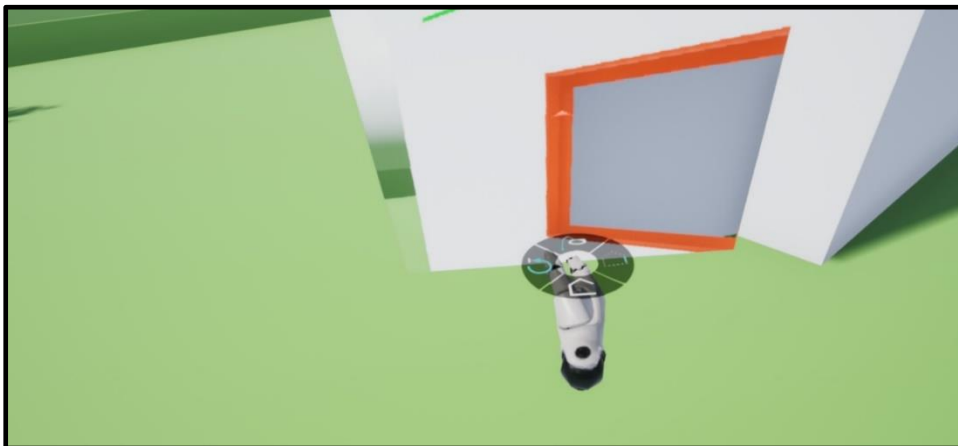


Abbildung 16: Geöffnete Lupe. Der ausgewählte Rahmen erscheint mit einem orangen Material

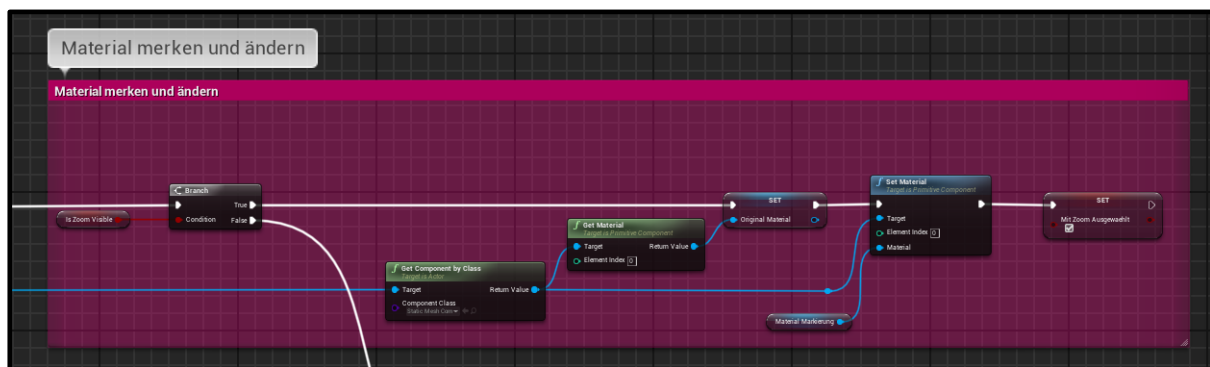


Abbildung 17: Implementierung der Materialspeicherung und Änderung

4. Problem: Kamera war nur an einen Motion Controller gebunden

Ursprünglich war „Camera Component“ im Motion Controller Blueprint in Abhängigkeit des Motion Controller Components implementiert (s. Abb. 17). Dies hatte jedoch zur Folge, dass die Kamera sich nur an den Motion Controller, der zuerst von den Sensoren wahrgenommen wurde, gebunden hat und somit auf der Lupe des anderen Controllers die „Sicht“ des Ersten zu sehen war und nicht wie gewünscht, die Sicht des Controllers, der den Zoom aktiviert hat.

Dieses Problem wurde behoben, indem die Kamera der Lupe im VR Pawn Blueprint als unabhängige Komponente implementiert wurde und diese per Code an den aktiven Controller gebunden wird (s. Abb.18 und 20).

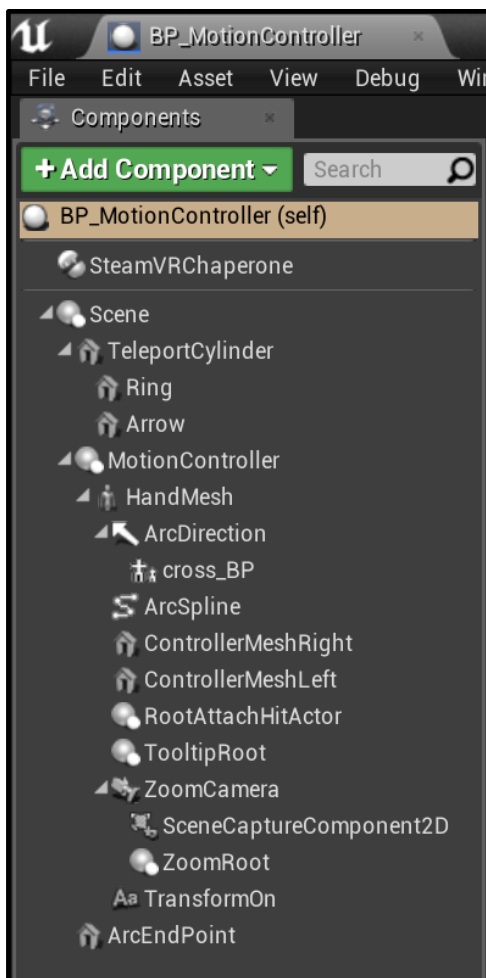


Abbildung 19: Vorher: Zoom-Camera abhängig vom Motion Controller

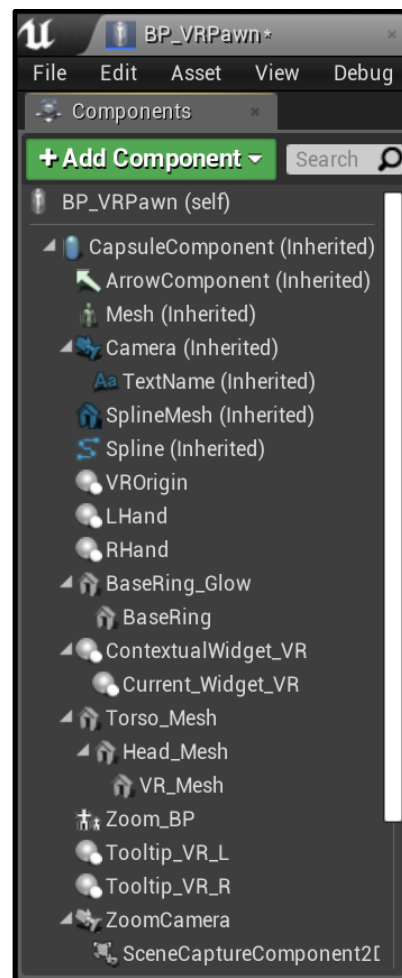


Abbildung 19: Nachher: ZoomCamera unabhängig von anderen Komponenten im VR-Pawn Blueprint.

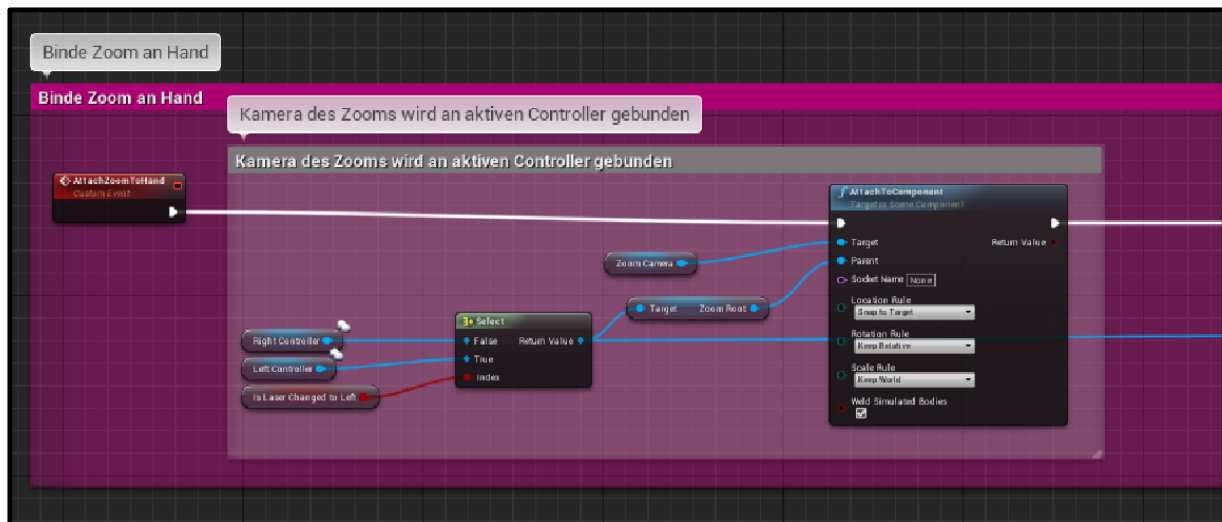


Abbildung 20: Implementierung der Kamera-Bindung an den Motion Controller, der denn Zoom gerade aktiviert hat

3. Problem: Mögliche Verwirrung durch Zoom-Plane

Bei einigen Probanden wurde eine Verwirrung bei Verwendung des Zooms festgestellt, da die Fläche keine klare Abgrenzung aufwies. Das Design wurde zu einer Lupe geändert. Aufgrund der verschiedenen Rotationen des linken und rechten Motion Controllers, wurde der Griff der Lupe weggelassen. Das Fadenkreuz zur leichteren Auswahl wurde optimiert (s. Abb. 21 und 22) .

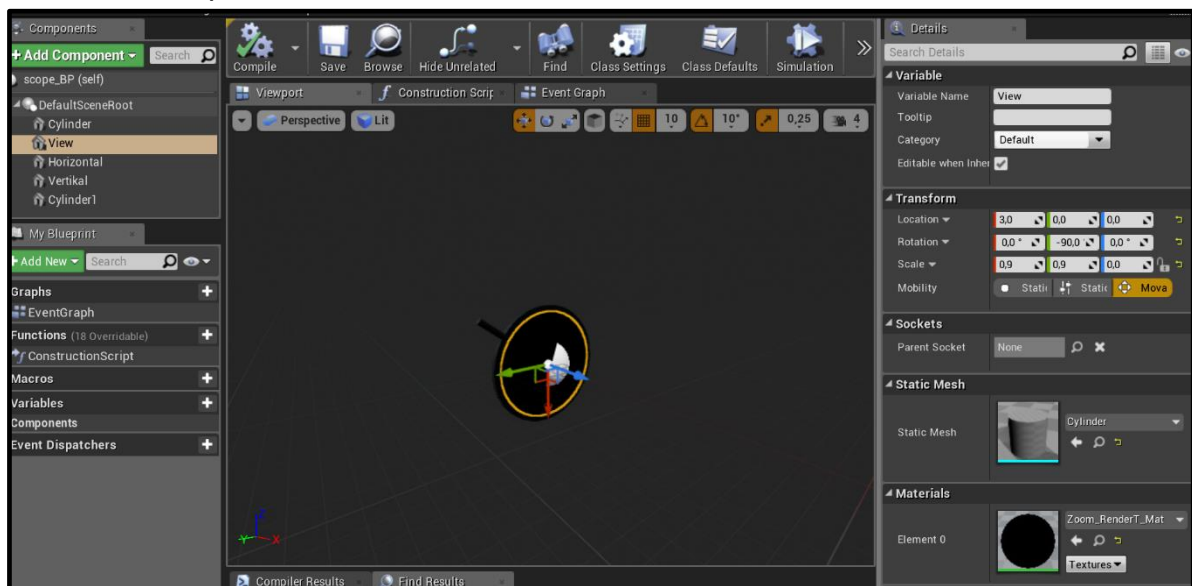


Abbildung 21: Erste Version der Lupe mit Griff



Abbildung 22: Geöffnete Lupe mit Fadenkreuz und ohne Griff im VR-Modus

6. Problem: Lupe entfernt Material des ausgewählten Objekts nach X-Ray und dessen Zurücksetzung

Es werden die zugehörigen Materialien aller Meshes zu Beginn gespeichert. Nach Verwendung des Zooms wird auf eine Map zurückgegriffen, um das ursprüngliche Material wiederherzustellen. Die Map hat einen Actor als Key und einer Struct mit einem Array als Value (siehe unten).

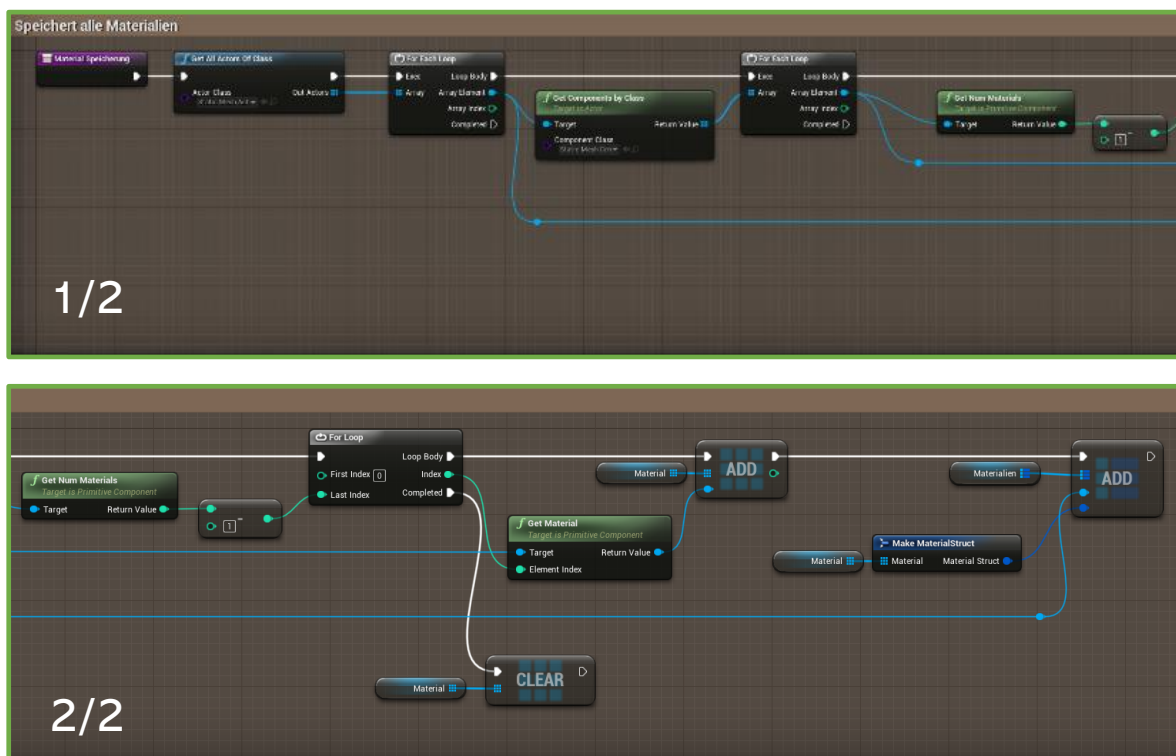


Abbildung 23: Material-Speicherung

Während des Ticks im VR-Pawn Blueprint werden die folgenden Funktionen während der Auswahl eines Meshes aufgerufen. Dies erfolgt nur bei aktivem Zoom.

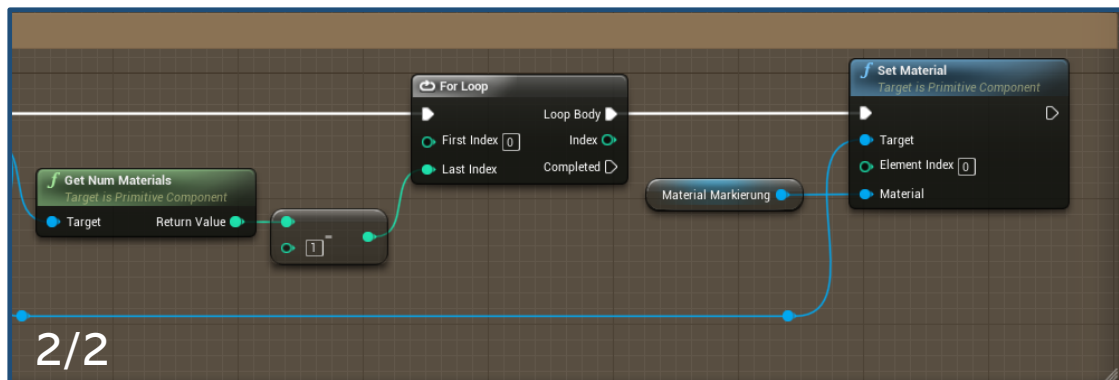
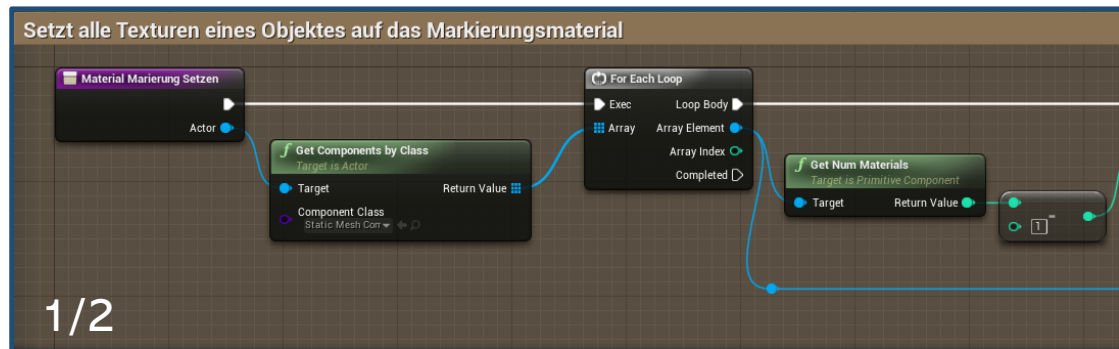


Abbildung 24: Material-Markierung setzen

Originales Material wird nach Deaktivierung des Zooms gesetzt:

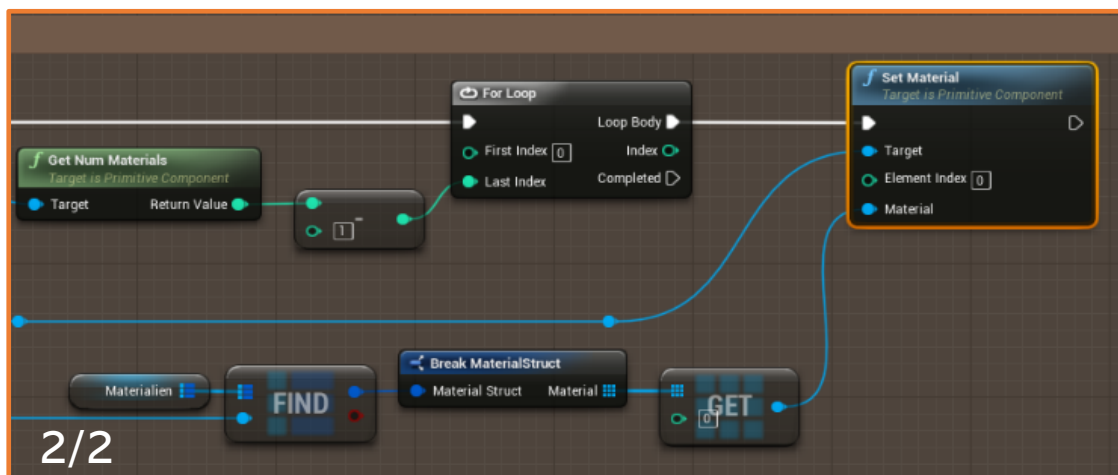
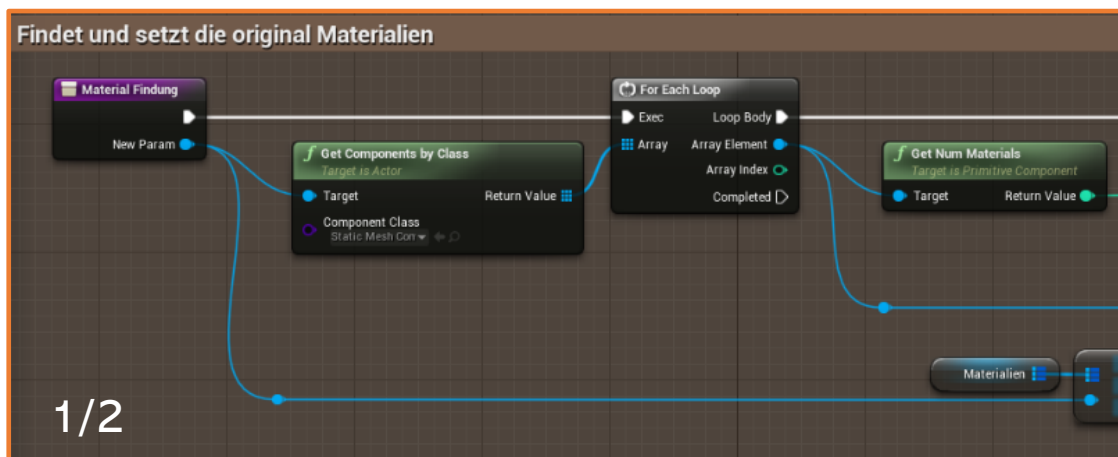


Abbildung 25: Material setzen

Implementierung des Zooms im VR-Pawn Blueprint:

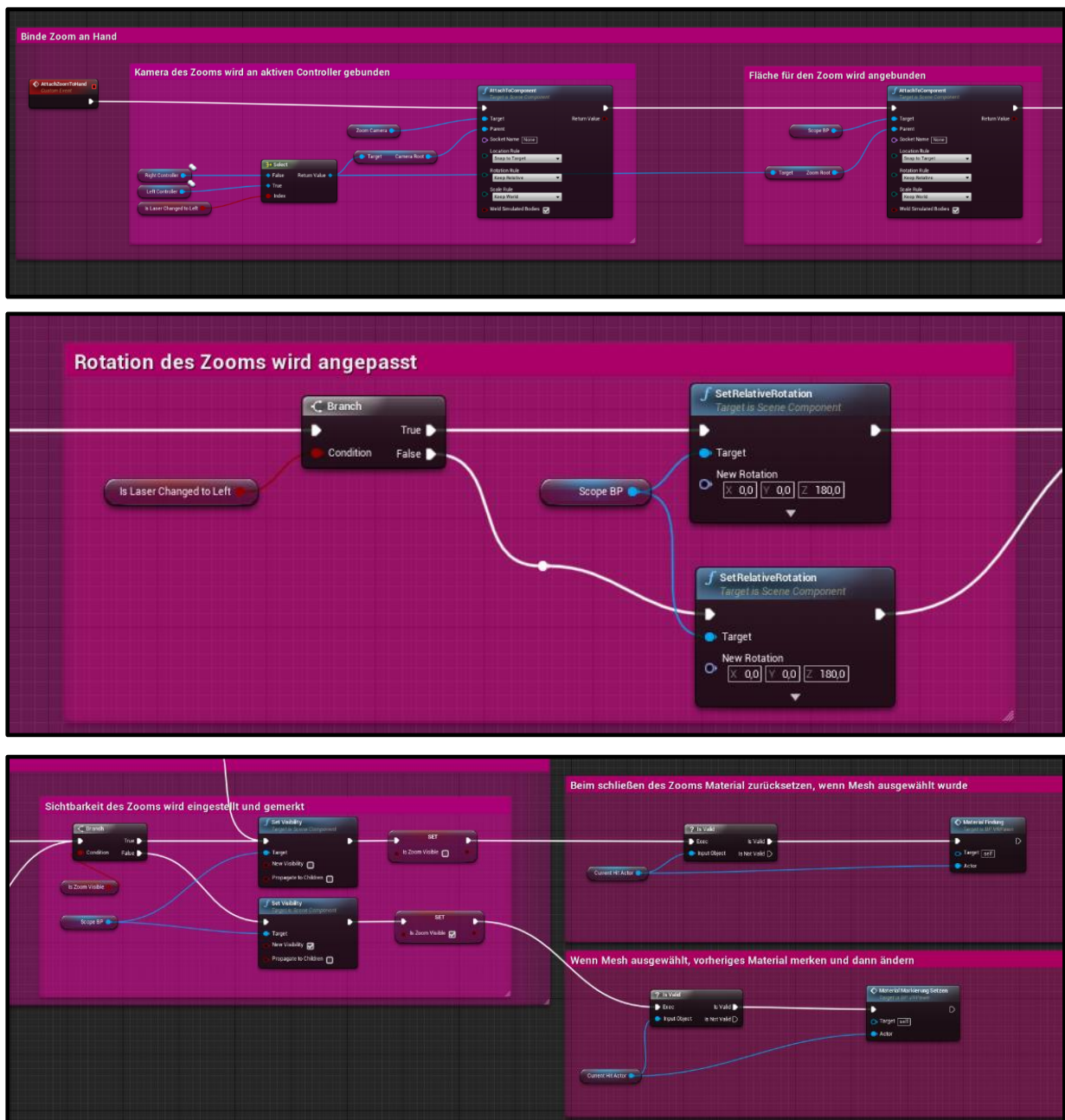


Abbildung 26: Implementierung des Zooms im VR-Pawn

Oculus Quest

Die Oculus Quest arbeitet anders als die HTC VIVE oder die Windows Mixed Reality, da es ein Standalone-Gerät ist. Das heißt die Oculus Quest benötigt keinen PC oder Konsole. Die Anwendung wird somit direkt in der VR-Brille aufgerufen. Deshalb wurde für die Oculus Quest ein eigenes Projekt erstellt, indem der Desktop-Modus nicht verfügbar ist.

1. Problem: Erste Tests mit Collab Viewer 4.23 sorgten für ständige Abstürze

- Collab Viewer 4.24 funktionierte deutlich besser
 - Keine Abstürze
 - Collab Viewer erkannte automatisch die Oculus Quest
 - Einfachere und mehr Auswahl zur Steuerung (direkt bei Settings/Input)
 - Es waren Oculus Touch Controller der Oculus Rift zu sehen

Aber Collab Viewer 4.24 unterscheidet sich sehr stark von dem alten Collab Viewer. Da unser Projekt in der Version 4.23 erstellt wurde und wir den Code von dem Collab Viewer 4.23 verwendeten, wäre das Migrieren der Blueprints in den Collab Viewer 4.24 zu aufwendig gewesen. Deshalb wurde nochmal versucht unser Projekt mit der Quest zu implementieren. Durch Verändern des „Event Server Posses Pawn“ im „Player Controller“ Blueprint ließ sich das Projekt erfolgreich starten.

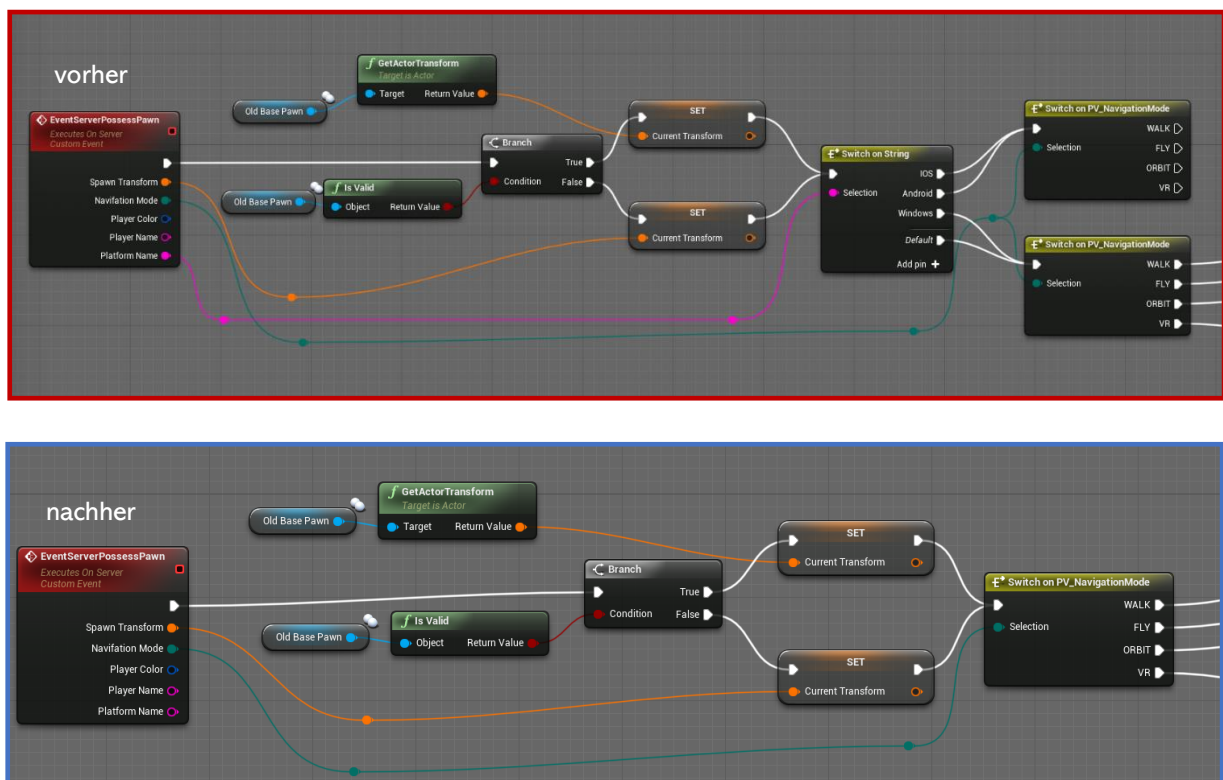


Abbildung 27: Ändern des Event Server Posses Pawns

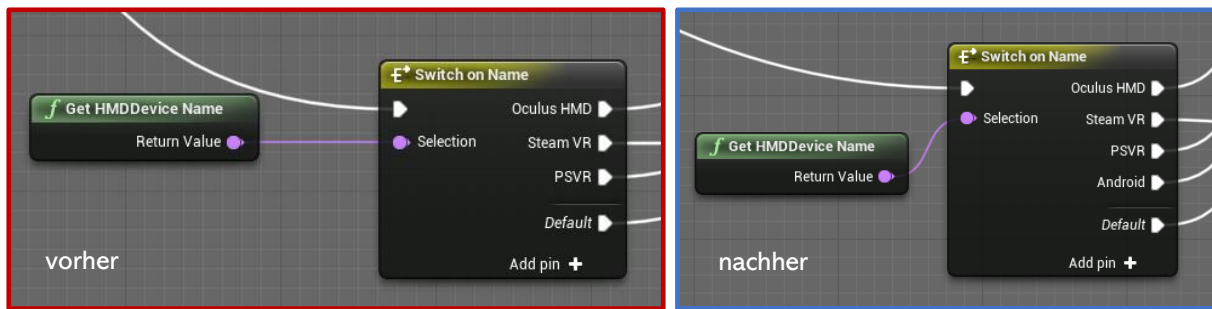


Abbildung 28: Android ist das Betriebssystem von der Oculus Quest. Deshalb wurde dies noch ergänzt.

2. Problem: Beim Auswählen eines Meshes fehlt die zugehörige Markierung.

Nach vielen Tests hat sich herausgestellt, dass die Oculus Quest die Custom Depth Funktion komplett ignoriert. Da bisher keine Lösung gefunden wurde, wird stattdessen ein neues Material als Auswahl benutzt. Das heißt das getroffene Mesh soll ein anderes Material haben und beim erneuten Anklicken oder beim Anklicken eines anderen Meshes, sich wieder zurücksetzen

3. Problem: Projektabstürze bei kleinsten Änderungen.

Unreal Engine 4 reagierte sehr empfindlich bei kleinsten oder sogar keinen Veränderungen im Viewport. Nachdem man „Compile“, „speichern“ und „Build“ ausführte, stürzte das Projekt beim „Launch“ mit der Oculus Quest sofort ab. Am Anfang war das Problem sehr irreführend, da der Fehler nicht zu identifizieren war.

Gelöst wurde das Problem, indem man das Projekt neu starten lies. Danach funktionierte das Launchen einwandfrei.

Automatische Texturierung

Vorgehen

Bei dem Starten der Anwendung wird für bestimmte IFC-Objektklassen über alle zugehörigen Meshes iteriert und passende Texturen zugewiesen.

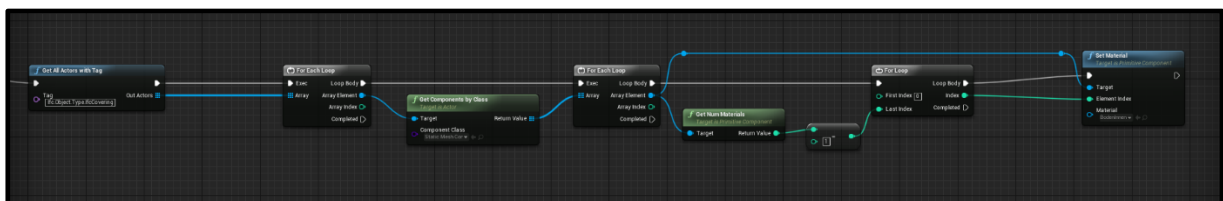


Abbildung 30: Über Texturen iterieren und passende zuweisen

1. Problem

Jedem Mesh kann automatisch nur eine Textur zugewiesen werden. Das liegt daran das ein Mesh entweder nur eine Textur hat, oder bei mehreren Texturen die Reihenfolge der Texturen nicht einheitlich ist. Dadurch kann z.B. einer Wand nicht automatisch unterschiedliche Texturen für Innen und Außen gegeben werden.

2. Problem

Türrahmen überschneiden sich mit Wanddurchbrüchen was bei unterschiedlichen Texturen Flackern hervorruft.



Abbildung 31: Türrahmen Überschneidung

Um dieses Problem zu beheben werden Türen um einen minimalen, nicht sichtbaren, Wert verkleinert und passend positioniert.

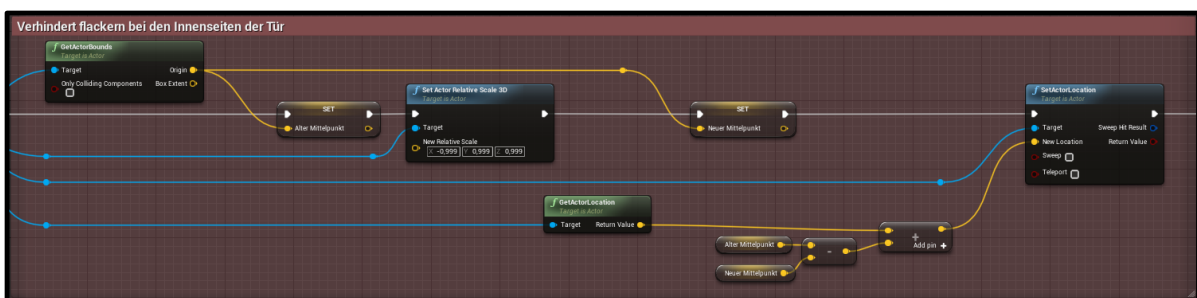


Abbildung 32: Türen verkleinern und positionieren

Verantwortlichkeiten

Es erfolgte eine Aufteilung in zwei Gruppen.

Gruppe UE4 (Julia Danko, Kevin Drabinski, Vaceslav Klepzov)

Die UE4-Gruppe war verantwortlich für die Anpassung des Collab Viewer Templates an unser Projekt.

Aufgaben:

- Entfernen von nicht notwendigen Blueprints
- Anpassung der Steuerung für HTC VIVE und Desktop
- Korrekter Import des Gebäudes
- Testen des Multi-Users
- Lösen der vorhandenen Fehler im Collab Viewer Template
- Anpassen und Erstellen der Tools

Gruppe IFC (Kevin Broy, Bernd Adamczyk)

Die IFC-Gruppe war verantwortlich für den Import der IFC-Metadaten nach UE4. Nachdem der Import erfolgreich funktionierte, arbeiteten die beiden Gruppenmitglieder an der Anpassung des Projekts.

Aufgaben:

- Entwicklung eines Programms, welches dazu beiträgt, die Metadaten aus der IFC-Datei zu einer für UE4 lesbare Datei aufzubereiten
- Darstellung der erstellten Metadaten im UE4 Projekt
- Erstellung der Oberfläche für den Zugriff der Metadaten im UE4 Projekt
- Auswahl der Metadaten, die in der Anwendung angezeigt werden sollen

Bernd Adamczyk

Verantwortlich für:

- Gestaltung und Umsetzung
 - Metadatenanzeige
 - Ingame-Menü
 - Tooltip
- Lesen und Darstellen der exportierten Datei mit den Metadaten auf der Metadatenanzeige
- Implementierung der Metadatenanzeige, des Ingame-Menüs und des Tooltip für die HTC VIVE Controller und für den Desktop-Modus
- Fehlerbehandlung
- Dokumentation
 - Diagramme
 - Protokollierung

Kevin Broy

Verantwortlich für:

- Erstellung eines Programms, welches dazu beiträgt, die Metadaten aus der IFC-Datei zu einer für UE4 lesbare Datei mit Java aufzubereiten
- Arbeit am Problem mit dem Multi-User zwischen VR-Brillen
- Funktion „Zurücksetzen“
- Import des Gebäudes in das UE4 Projekt
- Metadatenspezialist für Prof. Dr. Fieberg
- Fehlerbehandlung
- Dokumentation
 - Anleitung
 - Technische Herausforderungen
 - Schwerwiegende technische Entscheidungen
 - Produktbeschreibung
 - Videos (Langfassung und Kurzfassung)
 - Pflichtenheft
 - Diagramme

Julia Danko

Verantwortlich für:

- Erstellung und Optimierung des Tools „Zoom“
- Gestaltung und Umsetzung
 - Startmenü
 - Logo
 - Anleitung (mit HTML und CSS)
- Arbeit am Problem mit der Markierung für die Oculus Quest und „Zoom“
- Arbeit am Problem mit dem Multi-User zwischen VR-Brillen
- Fehlerbehandlung
- Dokumentation
 - Pflichtenheft
 - Lastenheft
 - Logbuch
 - Anleitung
 - Projektdurchführung
 - Technische Herausforderungen
 - Protokollierung

Kevin Drabinski

Verantwortlich für:

- Dynamische Materialänderung
- Texturierung der Karte
- Import des Gebäudes in das UE4 Projekt
- Anzeige der Tastenbelegung im Desktop-Modus in der Anwendung
- Optimierung der Lösung für das Zoom Material Problem
- Entfernung von nicht notwendigen Blueprints
- Strukturieren von Blueprints
- Fehlerbehandlung
- Dokumentation
 - Technische Herausforderungen

Vaceslav Klepzov

Verantwortlich für:

- Koordinierende Aufgaben
- Kontaktperson für Prof. Dr. Lux und Prof. Dr. Fieberg
- Belegung der Steuerung für den Desktop-Modus, HTC VIVE und Oculus Quest Controller
- Implementierung der Oculus Quest in das Projekt
- Anpassen des Transform-Modus
- Funktion „Zurücksetzen“
- Import des Gebäudes in das UE4 Projekt
- Fehlerbehandlung
- Dokumentation
 - Pflichtenheft
 - Abweichungen vom Pflichtenheft
 - Anleitung
 - Steuerung
 - Technische Herausforderungen
 - Schwerwiegende Designentscheidungen
 - Videos (Langfassung und Kurzfassung)
 - Protokollierung

Logbuch

Direkter PDF-Downloadlink über Redmine: [Logbuch](#)

Videos

Kurzfassung

YouTube Link: [Kurzfassung](#)

Langfassung (Making Of)

YouTube Link: [Langfassung](#)

Referenzen und Literatur

Literatur:

Borrmann, A., König, M., Koch, C., Beetz, J., (2015), Building Information Modeling. Technologische Grundlagen und industrielle Praxis, 1.Aufl., Wiesbaden.

Referenzen:

Redmine Wiki

Dokumentation Übersicht: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Dokumentation>

Unreal Engine 4: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/UE4>

IFC: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/IFC>

Java: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Java>

Multiplayer: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Multiplayer>

Zoom: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Zoom-Funktion>

Oculus Quest: http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Oculus_Quest

Material: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Material>

Charakter: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Charakter>

Blender: <http://host172.ik.w-hs.de/redmine/projects/spmisose2019facility1/wiki/Blender>

Weiteres

Projektmanagement Präsentation

Direkter PDF-Downloadlink über Redmine: [Präsentation](#)

Show & Tell Plakat

Direkter PDF-Downloadlink über Redmine: [Plakat](#)